402-CD-001-002

EOSDIS Core System Project

System Integration and Test Plan for the ECS Project Volume 1: Interim Release 1 (IR-1)

Final

February 1995

Hughes Applied Information Systems
Landover, Maryland

System Integration and Test Plan for the ECS Project
Volume 1: Interim Release 1 (IR-1)

Final

February 1995

Prepared Under Contract NAS5-60000
CDRL Item 064

SUBMITTED BY

Mark Settle /s/	1 Feb. 1995
Marshall A. Caplan, Project Manager EOSDIS Core System Project	Date

Hughes Applied Information Systems
Landover, Maryland

This page intentionally left blank.

Preface

This document is a formal contract deliverable with an approval code 1. It requires Government review and approval prior to acceptance and use. Changes to this document shall be made by document change notice (DCN) or by complete revision.

This document is under ECS Project Configuration Control. Any questions should be addressed to:

Data Management Office
The ECS Project Office
Hughes Applied Information Systems
1616 McCormick Dr.
Landover, MD 20785

This page intentionally left blank.

Change Information Page

List of Effective Pages	
Page Number	**Issue**
Title	Final
iii through x	Final
1-1 and 1-2	Final
2-1 through 2-4	Final
3-1 through 3-14	Final
4-1 through 4-124	Final
A-1 through A-22	Final
AB-1 through AB-4	Final
GL-1 through GL-16	Final

Document History			
Document Number	**Status/Issue**	**Publication Date**	**CCR Number**
194-402-VE1-001	Original	June 1994	
402-CD-001-002	Final	February 1995	95-0037

This page intentionally left blank.

Contents

Preface

1. Introduction

2. Related Documents

3. System Integration and Test Process

4. Interim Release 1

Figures

This page intentionally left blank.

1. Introduction

1.1 Identification

This document is submitted as required by CDRL item 064, DID 402/VE1, whose requirements are specified in this document as a deliverable under the Earth Observing System Data and Information System (EOSDIS) Core System (ECS) contract (NAS5-60000).

1.2 Scope

The ECS System Integration and Test Plan of Interim Release 1 (IR-1) - Volume 1 (SITP) delineates the process for integrating the major ECS segments and elements and verifying that the ECS complies with the Functional and Performance Requirements Specification (F&PRS), and the Interface Requirements Documents (IRDs). It identifies a schedule for performing such activities, describes the need for resources and the responsible test organizations.

ECS Releases are keyed to mission support: Release IR-1 provides support to TRMM Early Interface Testing and Science Algorithm I&T. Release A provides support to TRMM Science Operations and TRMM Ground Systems Certification Testing. Release A also provides the functional capabilities needed to support early ESDIS Ground System Testing for the EOS AM-1 and Landsat 7 missions. Release B provides support to EOS AM-1 Mission Operations and Science Operations, and it provides support to ESDIS Ground System Certification Testing for the EOS AM-1 and Landsat 7 missions. Release B also provides archive and distribution services for the Landsat 7 and COLOR missions, and it provides product generation support for COLOR. Releases C & D provide evolutionary enhancements to the ECS services provided in the earlier Releases.

1.3 Purpose and Objectives

This test plan provides a road map to this phase of the verification process by providing a breakdown of the activities to be performed into manageable units called builds and threads. The test plans are, essentially, the written outline for the step-by-step test procedures which, when issued later, become the detailed instructions on how to perform the verification of the ECS system.

1.4 Status and Schedule

This version of the document is due two weeks prior to the Preliminary Design Review (PDR). As an approval code 1 document, the ECS System Integration and Test Plan document requires Government approval prior to its acceptance and use

This document discusses the System Integration process and a proposed Build Thread Plan for the ECS System as it will be delivered at Interim Release One (IR-1). The Build and Thread Tests to be performed, are described at a summary level as well as the corresponding test descriptions and test cases. Appendix A contains a matrix mapping of IR-1 test cases to the Functional & Performance Requirements Specification (F&PRS) for this release.

1.5 Document Organization

The document is organized into four chapters:

Section 1 Introduction, contains the identification, scope, purpose and objectives, status and schedule, and document organization.

Section 2 Related Documents, provides a bibliography of parent, applicable and reference documents for the System Integration and Test Plan.

Section 3 ECS Integration and Test Process, describes the process used to test and verify the ECS.

Section 4 Interim Release 1 System Integration and Testing, describes the specific system level thread and build tests, which will be used to verify the functionality of Interim Release 1.

Appendices, acronyms and a glossary are at the end of the document.

2. Related Documents

Figure 2-1 illustrates the relationships of The System Integration and Test Plan (SITP) to other ECS documents.

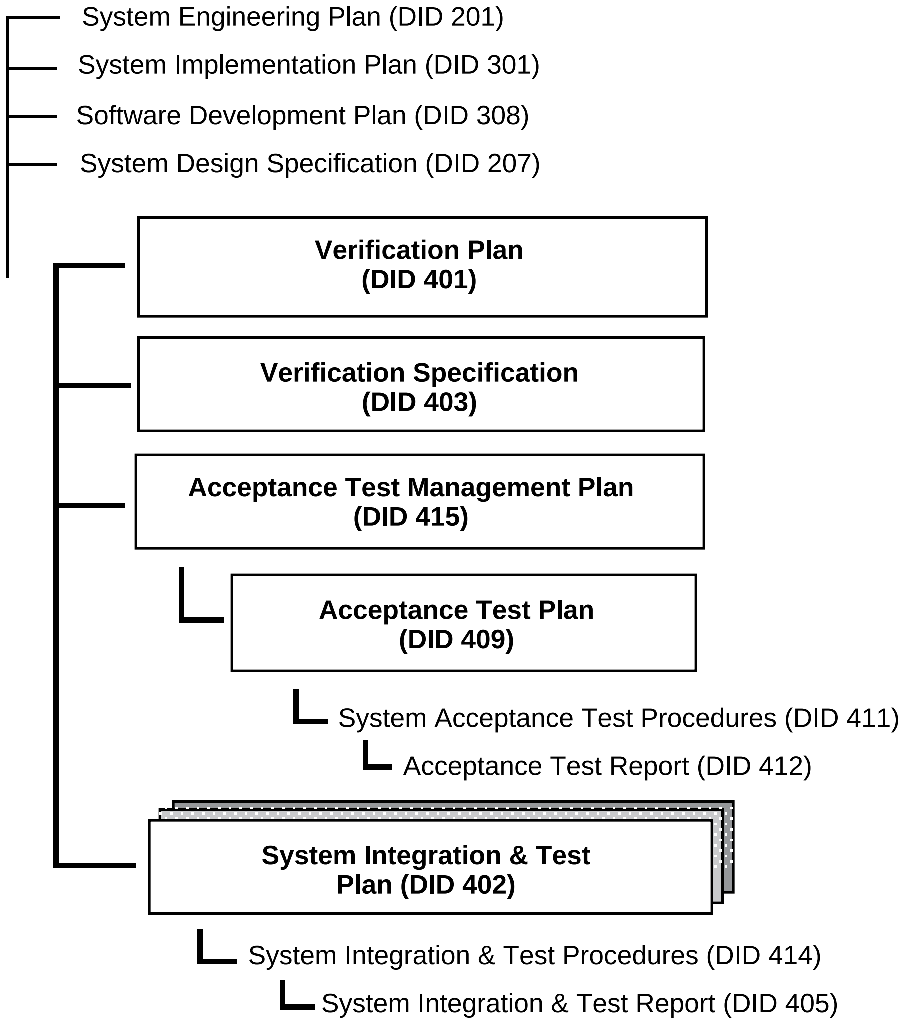
System Engineering Plan (DID 201)
System Implementation Plan (DID 301)
Software Development Plan (DID 308)
System Design Specification (DID 207)
Verification Plan (DID 401)
Verification Specification (DID 403)
Acceptance Test Management Plan (DID 415)
Acceptance Test Plan (DID 409)
System Acceptance Test Procedures (DID 411)
Acceptance Test Report (DID 412)
System Integration & Test Plan (DID 402)
System Integration & Test Procedures (DID 414)
System Integration & Test Report (DID 405)

Figure 2-1. System Integration & Test Document Relationships

2.1 Parent Documents

The following documents are parent to the test processes and procedures addressed in this document. In the event of any conflict between any of these documents and this document, the parent document(s) shall take precedence.

194-201-SE1-001	Systems Engineering Plan for the ECS Project
194-207-SE1-001	System Design Specification for the ECS Project
194-301-DV1-002	System Implementation Plan for the ECS Project
194-308-DV2-001	Software Development Plan for the ECS Project
420-05-03	Earth Observing System (EOS) Performance Assurance Requirements for the EOSDIS Core System (ECS)
423-41-01	Goddard Space Flight Center, EOSDIS Core System Statement of Work
423-41-02	Goddard Space Flight Center, Functional and Performance Requirements Specification for the Earth Observing System Data and Information System (EOSDIS) Core System
423-41-03	EOSDIS Core System Contract Data Requirements Document

2.2 Applicable Documents

The following documents are directly applicable to this plan. In the event of conflict between any of these documents and this plan, this System Integration and Test Plan will take precedence.

101-101-MG1-001	Project Management Plan for the EOSDIS Core System
193-103-MG3-001	Configuration Management Procedures for the ECS Project
193-105-MG3-001	Data Management Procedures for the ECS Project
193-203-SE1-001	User Interface Requirements Study for the ECS Project, Outline
194-206-SE2-001	Version 0 Analysis Report
194-208-SE1-001	Methodology for Definition of External Interfaces for the ECS Project
194-401-VE1-002	Verification Plan for the ECS Project, Final
194-403-VE1-002	Verification Specification for the ECS Project, Final
194-409-VE1-001	Overall System Acceptance Test Plan for the ECS Project
194-415-VE1-002	Acceptance Testing Management Plan for the ECS Project, Final
194-501-PA1-001	Performance Assurance Implementation Plan for the ECS Project

2.3 Information Documents

The following documents, although not directly applicable, amplify or clarify the information presented in this document, but are not binding.

194-219-SE1-001	Interface Requirements Document Between EOSDIS Core System (ECS) and the NASA Science Internet (NSI)
194-219-SE1-003	Interface Requirements Document Between EOSDIS Core System (ECS) and Landsat 7 System, Working Draft
194-219-SE1-005	Interface Requirements Document Between EOSDIS Core System (ECS) and Science Computing Facilities
194-219-SE1-006	Interface Requirements Document Between EOSDIS Core System (ECS) and Affiliated Data Centers, Preliminary
193-219-SE1-008	Interface Requirements Document Between EOSDIS Core System (ECS) and Program Support Communications Network, Draft
194-219-SE1-018	Interface Requirements Document Between EOSDIS Core System (ECS) and Tropical Rainfall Measuring Mission (TRMM) Ground System
194-219-SE1-019	Interface Requirements Document Between EOSDIS Core System (ECS) and Earth Observing System (EOS) AM-1 Flight Operations
194-219-SE1-020	Interface Requirements Document Between EOSDIS Core System (ECS) and NASA Institutional Support Systems

This page intentionally left blank.

3. System Integration and Test Process

This section defines the process used on the ECS Program to ensure thorough integration and verification of the ECS at the system level as defined by the Functional & Performance Requirements Specification (Level 3), Interface Requirements Documents and FOS Mission-specific Level 4 requirements.

The broad verification approach chosen for System Integration and Test, Build/Thread Testing, is explained. Within this context, the word "Testing" refers to "Test" in the broader context of "verification", e.g., "Test" encompasses the four activities of demonstration, analysis, inspection and test which comprise verification.

The subparagraphs that follow define the System Integration & Test process from three points of view. First, the philosophical approach, Build/Thread is described. Next, processes actually used to plan the testing and how those processes will be carried forward to complete the SI&T effort are discussed. Then, at a more detailed level, the day-to-day management of System Integration Testing is described in order to highlight the controls applied to the process. Finally, several special classes of testing required by the uniqueness of the ECS Project are discussed.

The processes described will be performed by the Integration and Test Team (I&TT) within the ECS contractor System Integration and Planning Office (SIP). Supporting this effort are the development segments, the Quality Office and the Configuration and Data Management Department.

Oversight of the System Integration and Test effort is provided by representatives of the Code 505 Integration and Operations office, the IV&V contractor and the ECS Project Independent Acceptance Test Organization.

3.1 System Build/Thread Test Approach

At the system integration and test level within the overall ECS verification process, a build/thread test approach has been chosen. Build/thread methodology relies heavily on the concept of a “thread” - the set of operational procedures, software and hardware that implement a function. Threads are tested individually to facilitate requirements verification and to simplify problem resolution.

The decomposition of release capabilities into threads allows flexible scheduling of development and testing. Such scheduling is influenced by considerations such as:

- Thread Dependencies. For example, basic communication services will be used in most threads. By integrating communications services first, they are available in tested form for all subsequent threads. This also eliminates repetitive use of a communications simulator and reduces test tool costs. By this mechanism critical, core components also get tested in many threads increasing confidence.

- Level of Development Complexity. A thread that requires many custom components may be scheduled late to ensure adequate development time. Threads that contain a significant amount of COTS will be integrated early while development is in process. This allows more parallelism in the development and integration process.
- Contingency Management. If unforeseen problems arise, threads and builds can often be re-arranged so that overall progress continues. In typical big-bang integration approaches, the entire effort may be stalled.

Threads are allocated to releases. Each thread is intended to be tested in one release, so if only partial capabilities of a function is required in a release the function will be subdivided into two separate threads.

Successfully integrated components (software, hardware, and data) that execute threads are merged with other threads in a gradual buildup of system capabilities - a build. Build tests verify newly available functions in their expanded environment. Regression tests confirm that newly combined functions do not degrade service from previously integrated components.

Groups of builds are, in turn combined to form larger builds until the complete release has been integrated. The final build test consists of end-to-end activities that, while still functionally oriented, approach actual operational scenarios.

Interim Release 1 (IR-1) has been defined to provide early support capability at a limited number of sites. From a build/thread point of view it behaves like any other, full-featured release. It has a final build that moves the system from the I&T process to the deployment process.

3.2 How Testing Is Planned

While Build/Thread is a philosophical approach to testing, this SI&T Plan is a specific application of that approach to the evolving ECS. This section will explain some of that process and chart the path ahead to complete the process for Release A and the subsequent releases. The intent here is to highlight the test-specific activities actually performed rather than to re-document the entire system engineering and verification process.

Initial Build/Thread design began with participation in the formulation of the Release Plan White paper. During this process external mission drivers were examined in order to understand the minimal functionality that had to be present for IR-1, Release A and Release B. This process was iterative in that more information became available with the passage of time. Additionally, initial cuts at functionality provided vehicles for better communication and, hence, understanding of the relationship of mission drivers. Co-incident with the development of the Release Plan a concerted requirements analysis and allocation effort was undertaken. This gave the test team a renewed appreciation for the requirements baseline. Co-incident with this effort the system architecture group began producing white papers explaining the new ECS architecture. These progressed into the individual Logical Object Models (LOMs) which describe the allocation of services at the logical level.

Using the information base thus gathered, suggestions for threads were formulated on the basis of the functions defined in Table 10 of the Release Plan. These were examined in the light of the criteria discussed in Section 3.1. Based on this, initial aggregations of the threads into builds and so were made and discussed with representatives from the development organizations to gather their assessment of reasonableness and correctness.

After several iterations, preparation of the written descriptions of the threads and builds was begun. The act of concisely describing the thread and the verification of it ends up being a fundamental part of the engineering process. In some cases, attempting this proved that the thread content was poorly chosen in that there may be too many unsatisfied external dependencies or that the results of the threads actions aren't sufficiently externalized to allow verifications. Such discoveries, caused revision to the build/thread plan resulting in the version of the plan submitted at SDR/RIR.

Detailed planning for the test takes place during the period between RIR and PDR/IDR. Any changes necessary as result of the review are incorporated. As the preliminary system design takes shape, the test organization monitors the process and continually re-validates the test plan. Changes proposed by the development organizations are reviewed and assessed as to their impact on the overall integration activity. Negotiation takes place. If mutually agreeable resolution can not be achieved, issues are elevated to management.

Each thread and build in the document is taken to the final level of detail by enhancing the description of the test flows to take into account the details of the implementation of the design. If the size of the thread warrants, multiple test *sequences* within a thread may be created. Sequences serve to enhance management of the testing activity by dividing the effort into smaller tasks that can be prepared and executed independently.

Given the increasing level of detail in the design of the release, better knowledge of the interfaces between components is available. This allows individual threads to be assessed for their ability to standalone during test. If because of design progress, a thread is found to have unexpected dependencies that cannot be met, two options exist. First, restructuring of a portion of the build/thread plan can occur. Because of the nature of the approach this can often be done with only minor impact on the overall effort. The second option is to provide test tools (stubs) to meet the needs of the thread. Early in the integration process, such stubs are more numerous because many of the threads are dedicated to provide mainly infrastructure.

Since a thread, or a sequence within a thread, defines a set of actions to be performed to evoke a desired response, the next step in the detailed planning process is to determine the range of test values to be used to exercise the function. A set of inputs is chosen to assess mainstream functionality. This is supplemented by values at the edge of the acceptable range and values beyond the acceptable range. For each set of input conditions, an expected set of outputs is postulated. The pairing of a set of inputs and outputs to be applied to a thread or sequence is considered to be a *test case*. Sufficient test cases are formulated to achieve certainty of correctness commensurate with the criticality of the function under test.

When this is accomplished for all builds and threads, this plan is submitted for PDR/IDR. With this, the end of the test planning phase occurs and test procedure preparation begins.

The next stage in the test process is the generation of detailed test procedures (DID 414/VE1) based on the plan. The PDR version of the plan essentially forms an outline for the test procedures. The test procedurea are a step-by-step set of instructions for the actual execution of the testing required to integrate and verify the thread. Additionally, the procedures have a concise definition of the test environment for the activity. Like the test plan, the procedures are developed in stages as more and more information becomes available during the development life cycle. Before code first becomes available after CDR, detailed design information and initial user manuals are used to begin the procedures. Another key activity during the period is the development of test tools and the collection/production of test data.

As software becomes available, initial procedures are checked out by execution against the developed system. Such execution both attempts to exercise the system and the procedures themselves. Problems that occur can be the result of either errors in the procedures or errors in the integration of the product. The test team members work with developers to investigate problems and find remedies. This cooperative process continues until the discrepancy rate moderates and until there is confidence that the procedures are correct. This generally occurs before the system under test is complete and performing correctly. The test procedures are delivered at this point in the progress.

At this point, test conduct begins. Test conduct is the execution of the test procedures against a software baseline that is under configuration control. The goal of conduct is the verification of requirements through successful execution of the test. During this time, portions of procedures may be executed out of sequence to concentrate on particular parts of the system. Test conduct continues until all parts of all procedures have been successfully executed.

Test conduct culminates with the formal execution of the procedure as a whole before appropriate witnesses. Required witnesses include representatives of the Quality Office, ECS Project Management and ESDIS Integration Office. As always, the authority to witness may be delegated or waived on a case-by-case basis.

At the completion of the formal execution of the System Integration Test, a Consent to Ship Review (CSR) is held. The CSR determines if the current activity of SI&T, has been completed successfully. The CSR is a formal meeting chaired by the Test Lead at which the following are presented:

- Initial Test Results - Based on a quick-look analysis of test data the outcome of the testing is presented.
- Deviations from Test Procedures - If during the demonstration, any deviations from the printed procedures were necessary, they are explained and discussed to establish that they did not invalidate the test execution.
- Non-conformance Report (NCR) Status - The status of all open NCRs is presented along with recommendations for their disposition
- Configuration Management Status - The CM organization reports on the status of the product baseline.
- Recommendation - Based on the previously presented material the Test Lead recommends that the acceptance/rejection of the test.

3.3 How Testing Is Controlled

This section discusses the basic, day-to-day management of the verification process. The fundamental steps to be described here are performed during test preparation, conduct and the formal demonstration. As the test progresses and matures the degree of formality and the frequency of process checks increases to insure timely completion and thorough testing. For the sake of this presentation, the activities of a theoretical "day" of testing will be described. It should be noted that this "day" is not necessarily an actual calendar day. During the preparation phase the unit of management might be a week, while during the formal demonstration it might actually be a day or even part of day.

A testing session begins with a pre-test meeting at which the planned activities are discussed, assignments made and the configuration for the test environment specified. The readiness of the environment to support the activity is assessed. Other than during critical times and during the formal demonstration, the pre-test meeting is held as the last part of the previous sessions post-test meeting.

The key to effective testing is the documentation or logging of what is done. Good note taking serves to:

- Capture the state of the product under test (e.g. version numbers, concurrent activities)
- Capture the chronology of actions - without this, reported problems cannot be recreated and understood to allow correction
- Provide an audit trail - this allows verification of coverage and collection/analysis of metrics.

At a minimum this information will be captured in individual test logs. Because the ECS is distributed, correlation of widely dispersed events is going to be especially difficult. Some thought is being given to construction of on-line tools to supplement hard copy notebooks and allow easier time correlation and collaborative testing. Extension to public domain products like NCSA Collage might be feasible.

The other key to the testing process is the control of the test environment. Included are software, hardware and "user/operational" environments. Software and hardware states are tracked and controlled by the CM system. During testing, the Test Lead is delegated authority from the Configuration Control Board to manage the test environment. The process used is discussed in the paragraphs on Non-conformance Reporting and Corrective Action (NCRCA). An area of difficulty in the distributed environment is the control of the user/operational environment. What is meant here is the need to understand, and, at critical times, control the actions of others within the test environment. During development and the early phases of test preparation, test and development will share the same hardware and system software environments. Aside from the impact on performance due to resource contention, updates to shared data, etc. can occur. This will be handled through close coordination with development. During the later stages of testing, the environment will be controlled by excluding other than test users, through strict access controls managed by the test lead.

Whenever problems occur they are entered into the ECS NCRCA. Unlike systems employing pre-screening, test policy is that all problems are entered into the NCRCA, assessed and dispositioned. While invariably duplicate and erroneous problems are entered, mechanisms are in place to quickly and efficiently disposition and close them out. This approach tends to insure that no problems are overlooked. Another benefit is our ability to collect metrics from the NCRCA system.

System Integration Testing maintains its own list of Non-conformance Reports (NCRs) within the NCRCA covering problems that occur during the time the product under test is controlled by SI&T. Thus, if, during preparation, a test team member discovers a problem, it is entered into the NCRCA as a System I&T problem. If during assessment (discussed later), it is determined to be caused by a component that has not yet been turned over to SI&T, the problem is transferred to the Segment Integration and Test portion of the NCRCA for resolution.

During all but the formal conduct portion of the System Integration & Test process, problem isolation and investigation is encouraged. This means that the problem is pursued in order to accomplish the integration of the system. Test personnel are expected to have the capabilities to perform the initial part of this effort. When this is insufficient development support is called upon. Information thus obtained is added to the NCR as an aid to disposition of the problem.

At the end of each test "day", a post test meeting is held by the test lead. Attending are test team members, development support personnel and Quality Office representatives (if they desire to attend). The purpose of the meeting is to assess progress, plan future activities and review the status of NCRs. This review includes:

- New NCRs - Each new NCR is presented (usually by the author). The problem encountered is described along with the author's recommendation for priority. The test lead assesses the NCR and assigns a disposition. Most often at this stage in the process the problem is assigned to a development representative for further investigation. The test lead assigns a priority.
- Open NCR Status - Status is sought on open NCRs. The developer support returns information of the problems that are being fixed in support of the testing. When a fix is completed, it is reported and the status of the NCR updated to "Fixed". This status means that the developer asserts the problem is solved. Before the NCR may be closed, two important things must happen. First, the Test Lead must allow the fix into the test baseline via the CM system. On a case-by-case basis, proposed fixes are considered for priority and impact on the test as a whole to determine the strategy for their inclusion. Some fixes are held until a related group is ready, others may be incorporated individually. Second, once the fix has been installed into the test baseline a test team member must verify the fix (usually by rerunning the activity described in the NCR). If this testing is successful, the NCR is closed at the discretion of the Test Lead.

3.4 Roles and Responsibilities

This section describes the responsibilities of different organizations in the integration and verification of the ECS.

3.4.1 System Integration and Test

The Systems Integration and Test (SI&T) organization is part of the System Integration and Planning (SI&P) Office which additionally has responsibility for Engineering Planning, Technical Assessment and Standards, V0 and External Activities, Interface Engineering, Co-ordination and Control, and Performance and Cost Modeling.

Figure 3-1 shows the ECS contractor's System Integration and Planning (SI&P) Office and the (SI&T).

The Integration and Test Team (I&TT) is primarily responsible for the ECS system integration and verification. Upon successful completion of testing, the I&TT will hold Consent to Ship Review (CSR) for delivering the software to the Independent Acceptance Organization Test (IATO) via the CM organization.

The Independent Acceptance Test Organization (IATO) is primarily responsible for the systems acceptance testing, which occurs subsequent to the system integration and test activities. Implied in this responsibility is the installation of the releases at the remote sites. Support for the Independent Verification & Validation (IV&V) contractor is managed via the IATO.

3.4.2 Configuration Management

All deliverable application software and software test tools will be controlled via the ECS Configuration Management. tool (ClearCase) which is administered and controlled by the Configuration Management Organization. During the entire development process the CM organization provides the software library function. As the SW product moves from development, to Segment I&T, to System I&T and ultimately to Acceptance testing, it is within the CM tool. The control authority over the baseline is delegated to the appropriate individuals during the process. In the case of System I&T, the Test Lead has authority to control changes to the CM baseline under test. Test Procedures and data used during the test will also be baselined and controlled.

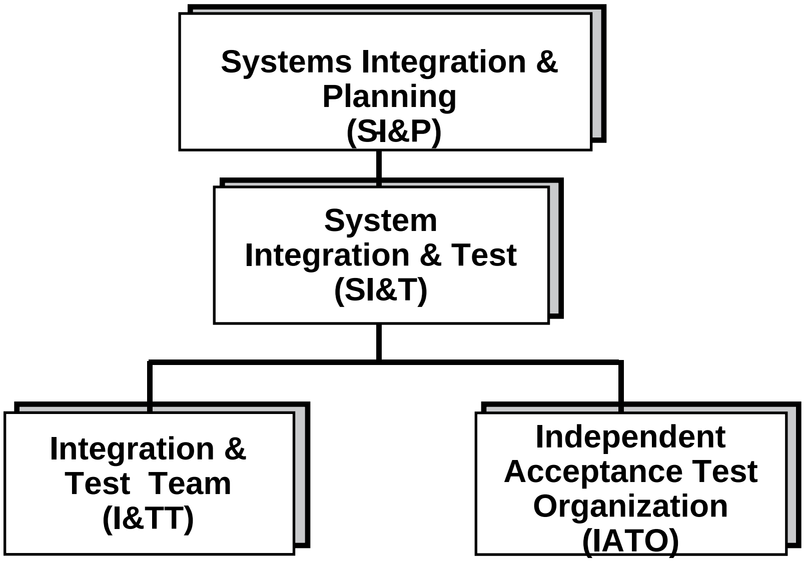
Systems Integration & Planning (SI&P)
System Integration & Test (SI&T)
Integration & Test Team (I&TT)
Independent Acceptance Test Organization (IATO)

Figure 3-1. Organizational Chart

3.4.3 Quality Assurance

A Non-conformance Reporting and Corrective Action (NCRCA) system managed by the Quality Assurance Office is used to control discrepancies identified in both documentation and software. A Non-Conformance Report (NCR), is used for any departure from design, performance, testing or handling requirements that affects hardware, software or documentation. The process for Non-conformance Reporting was described in Section 3.3.

The Quality Office oversees many facets of the testing process through inspections of work in progress. At formal demonstrations the Quality Office witnesses the test activities to insure compliance with written procedures. Quality Office responsibilities are described in the Performance Assurance Implementation Plan (PAIP), DID 501/PA1.

3.5 System I&T in a Multi-Track Environment

For a specific release, two main development processes will be used: the Formal Development Process and the Incremental Development Process described in the Multi-Track Development White paper. Figure 3-2 from that document shows the relationship between the two processes.

Both formal and incremental development tracks will be implemented to: 1) assure compliance with acknowledged requirements, 2) provide traceability of requirements allocation to tracks, 3) implement an integration process that brings the separately developed pieces together into an integrated whole, and 4) provide a process for control of interfaces that supports integration.

From a System Integration & Test point of view, the Incremental Path can be thought of as nothing more than another development methodology. As currently constituted, software from this path enters the test process at the Test Readiness Review (TRR) for integration at the segment level. This means that it has rejoined the normal development flow and should be indistinguishable from formally developed software.

In practice, software from the incremental path is different. First, because its requirements analysis and objectives determination process is handled out of the formal path, SI&T planning for the increments is less precise early in the formal process. The Integration team through participation with the segment test organizations in the planning and testing of the Evaluation Packages. This means that the I&T personnel have earlier contact with and influence over incrementally developed products than with formally developed ones. This knowledge and involvement means that we have the ability to react quickly to evolutionary changes during the incremental process. Any such changes will be reflected as updates to formal track test documentation, as necessary.

3.6 Integration and Interface Testing

The integration of elements and segments and the integration of the ECS with external systems is a fundamental part of the System I&T process. The interface requirements for the ECS will be documented in various interface requirements and control documents, both internal and external:

- Methodology for Definition of External Interfaces (DID 208/SE1)
- Interface Requirements Documents (DID 219/SE1)
- External Interface Control Documents (ICDs) (DID 209/SE1)
- ECS Internal ICDs (DID 313/DV3)

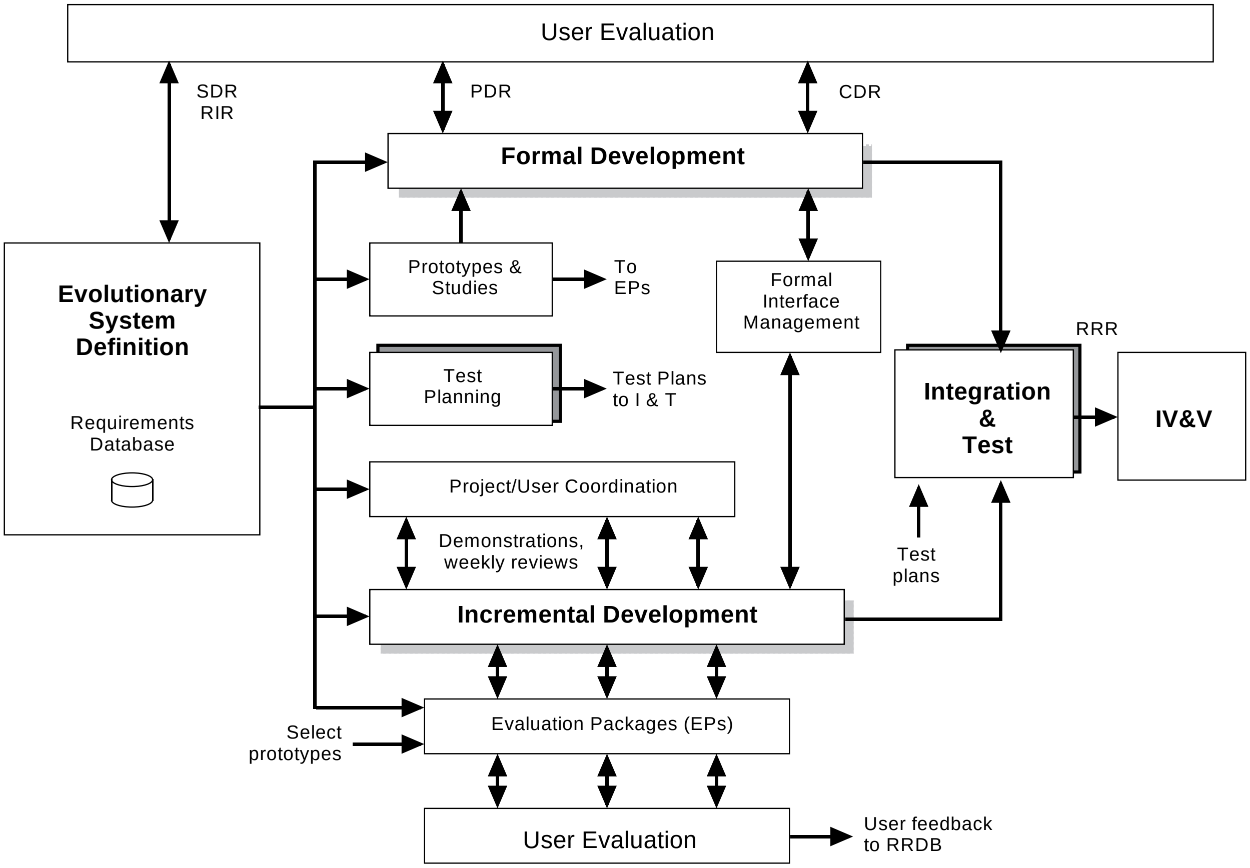

Figure 3-2. System I&T and Multi-Track Environment

Interfaces are exercised and tested as a normal part of the build/thread testing process. The build-up of the system described in this document is fundamentally one of aggregation of components (threads and builds). The glue are the defined internal interfaces. The tests, herein described, implicitly verify these interfaces. External interfaces are often the driving requirement for creation of threads and builds.

To perform the integration and interface verification, simulators and simulated data flows will be used until actual system capabilities exist on both sides of the interface. Version 0 data sets will be valuable as a set of early test data. For external interfaces it may be necessary to remain with simulators throughout the test process because of schedule (and scheduling dependencies). System Integration and Test is committed to performing engineering-level early interface testing with external systems whenever possible. Such activities will be arranged through the interface definition process and the Ground System Integration Working Group.

3.7 Test Tools

Four areas of applicability of test tools to the ECS Program have been identified. First, tools are needed to support automated test planning and management. These tools assist in the development and tracking of test cases, test data, test results, as well as the mapping of requirements to test cases. The Requirements and Traceability Management (RTM) tool has been selected by the ECS project to support the requirements management process. The RTM tool provides the means to record all relationships and dependencies between requirements, documentation, releases, services, and more specifically, test specifications. RTM assists systems engineers in defining requirements, assigning them to release, and mapping them to formal test cases. In addition, test results will be recorded and mapped to requirements within RTM, so that at any point, the status of the test and verification of a specific requirement can be checked.

Second, tools are needed to simulate interfaces, especially for systems external to ECS. External interfaces refer to systems outside of the scope of the ECS contract. The interfaces may be to systems already in existence, systems that are being built as part of the overall ESDIS project, or systems being built by other Government agencies or other countries. Simulators for external interfaces generate and transmit data streams in the identical format that represent the specifics of the real system's data stream.

Third, tools are needed to automatically execute test procedures or scenarios. Included in this are Remote Terminal Emulators (RTEs), to emulate live users, data generators, to generate simulated input data, and programmable test languages. For RTEs, a tool is needed that is capable of emulating the maximum number of users that ECS is required to support at one time. For data generators, simulated data sets (in Level 0 format) from each of the instruments and possibly from existing satellites will be needed. A programmable test language having some simple command structures and is capable of controlling interface simulators as well as the RTE is needed.

Finally, data reduction and analysis tools will be needed to process and summarize the large amount of test data anticipated by the ECS Program. Data reduction and analysis tools are utilities designed to analyze test output data, including utilities to compare test output to benchmark data. Some form of sophisticated file compare utility is needed to compare expected test results to actual test results. A data reduction utility is needed to reduce large amounts of output data to some meaningful evaluation of the data's quality.

For system integration and test, the bulk of the testing will be performed within the ECS Development Facility (EDF), located within the Hughes complex. Many simulated data streams and simulated functional capabilities will be used due to the evolving maturity of the ECS.

Since most of the tests at the EDF will have a large simulated component, it will be important to accurately record and be able to accurately reproduce test conditions, test data streams, test workloads, etc.. Test tools assist in managing and controlling the test environment to make reproducible and controllable test conditions possible.

3.8 Tool Kit Testing

Tool kits provide a controlled interface into the services provided by the ECS. These interfaces are provided at four levels: User, Software, Algorithm and Application Interface. Because these are fundamentally different, the testing done on each will be significantly different. These are discussed separately in the paragraphs that follow.

3.8.1 User Interface Tool Kit Testing

User Interface Tool Kits provide the functionality that a remote user must employ to access the user oriented services offered by the ECS. With this software the remote user has access to the full graphical interface allowing data search, data browse and data access. Other, simpler interfaces accomplished through remote login are also provided.

Testing of the user interface tool kits is largely indistinguishable from the normal mainline verification of the Release as a whole. The tool kit software is identical with that which will run in the local terminals at the EOC and the DAACs themselves. The only differences requiring specialized testing are the potential impact of the network connection and the degree of equipment compatibility. The ability of the tool kits to mitigate the impact of the network is verified by using a remote login through the network while testing is going on in the EDF. Through careful attention to message routing, messages can be directed out through long loops on the Internet even though the connection could be made directly on the local LAN.

3.8.2 Algorithm Interface (PGS) Tool Kit Testing

The PGS Tool Kits exist in two forms. The first is delivered to scientists at the Science Computing Facilities (SCFs) to provide an environment in which algorithms destined for operational use within the ECS Product Generation System (PGS) will be developed and tested. The second form of the PGS Tool Kit replaces the first when the SCF-developed algorithm actually executes in the PGS at a DAAC. These two cases are illustrated in Figure 3-3. Both tool kits provide identical calling sequences to the science algorithm.

The SCF version of the PGS Tool Kit as illustrated in Figure 3-4, is delivered before formal delivery of the releases to allow algorithm developers an early start. Thus, the tool kit is tested separately through use of a benchmark algorithm and test drivers.

Since there is no integration of the tool kit with other parts of the ECS, this testing is not accomplished as a part of System Integration Testing and is not described in this plan. In fact, SI&T cooperates with Segment I&T to perform SCF toolkit testing.

The operational version of the PGS Tool Kit as illustrated in Figure 3-5, is delivered as part of a complete release and is tested with that release. The tool kit is tested coincident with the testing of the PGS through the use of one or more benchmark algorithms.

The focus of this testing is two fold. First, it must prove that the interfaces to the algorithm meet the interface definitions of the PGS Tool Kit Specification. Second, it must prove that the PGS interfaces with the algorithm for the delivery/receipt of data and the scheduling/control of algorithms.

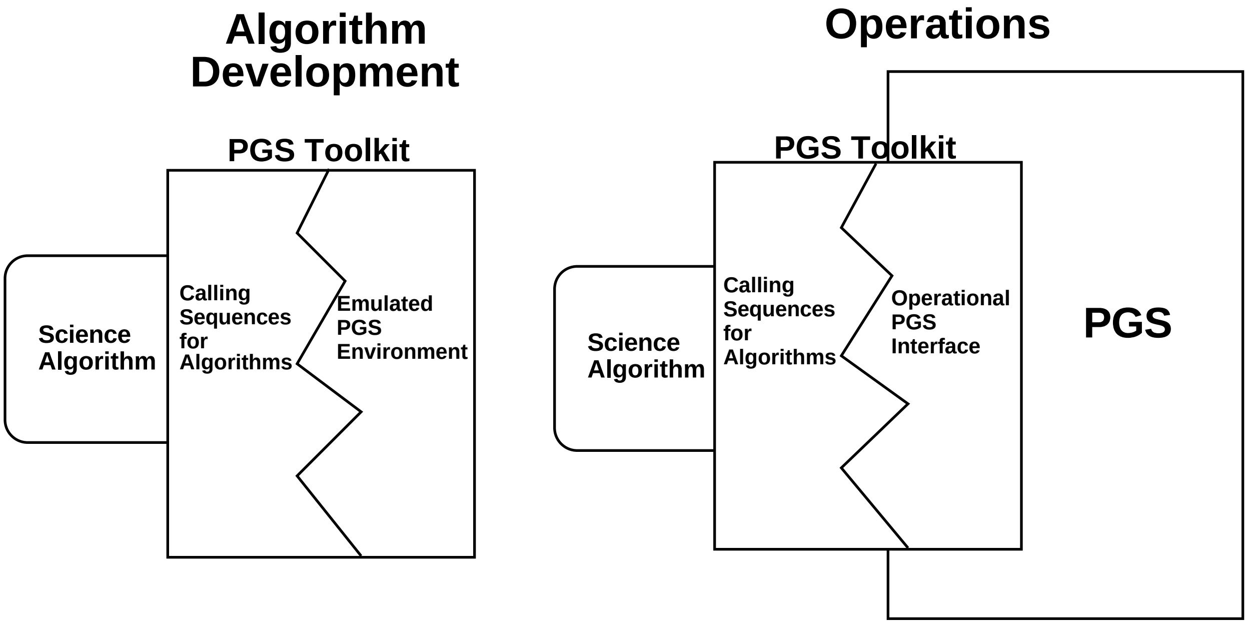

Figure 3-3. PGS Tool Kit Environments

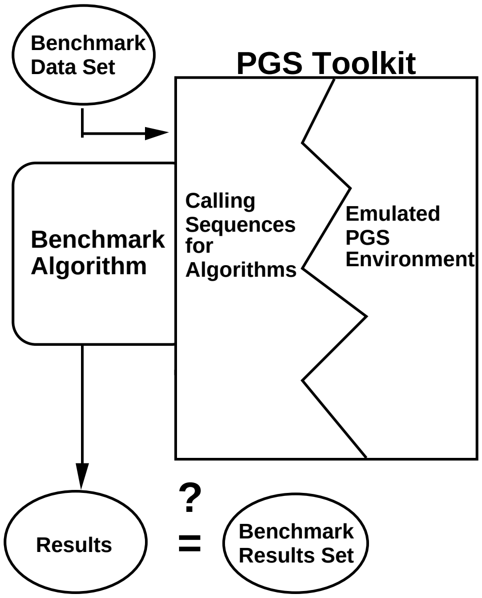

Figure 3-4. PGS SCF Tool Kit Test Environment

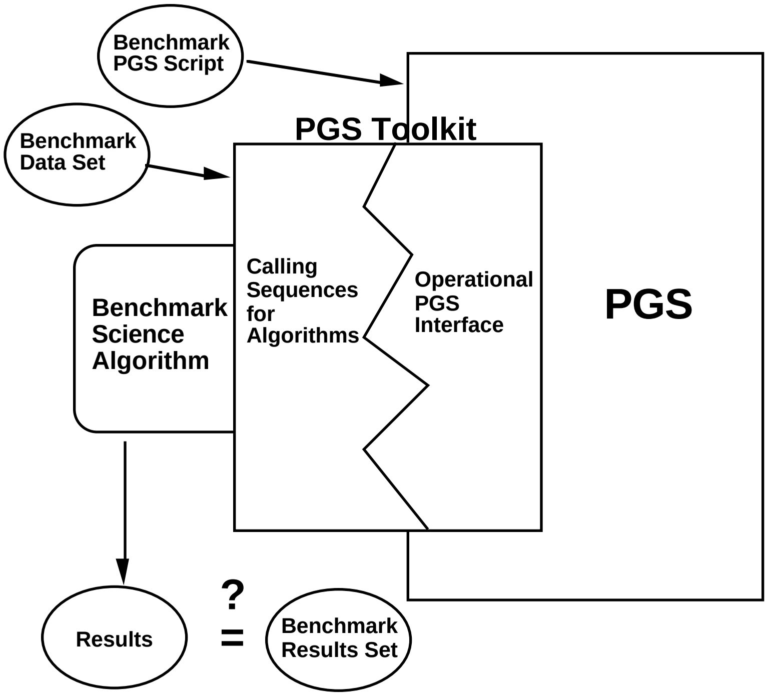
Benchmark PGS Script
Benchmark Data Set
PGS Toolkit
Benchmark Science Algorithm
Calling Sequences for Algorithms
Operational PGS Interface
PGS
Results
?
=
Benchmark Results Set

Figure 3-5. PGS Testing Environment

3.8.3 Applications Program Interface (API) Testing

Applications Program Interfaces (APIs) are provided by the ECS to allow properly authorized, externally generated software systems access to ECS Segment/Element functions at the client/server level. These interface codify the interfaces used within the ECS by its own components.

Testing of these APIs will involve the use of simple driver programs that validate compliance to the interface specifications. The operations attempted through the APIs will be chosen to explore the full range of the interface's parameters with emphasis on the most probable usage. The other necessary testing emphasis must be on verifying that the API protects the core ECS system from transgressions by the user processes.

3.9 Schedules and Dependencies

Detailed schedule information resides in the ECS Level 1 Master Schedule, (DID 107) and the notebook maintained by the Release Manager. As release planning continues, test scheduling will be refined and presented in subsequent updates to this document. In addition, other required documentation vehicles, such as the ECS Intermediate Logic Network, that are regularly released will be used as the vehicle to disseminate current schedules. The following Figure 3-6 shows the planned System Integration & Test activity for IR-1 as of PDR. Bars shaded as “Critical” are referring to the priority of the thread or build testing in relationship to its ETR delivery schedule,, duration of system level testing and/or degree of complexity of the testing to be performed.

Name	Duration in Workdays	Scheduled Start
ETR 1	1d	8/15/95
T1-2. ESN WAN Thread	11d	8/15/95
T1-1. DAAC LAN Thread	16d	8/15/95
B1. System Access Build	30d	9/15/95
ETR 3	1d	10/16/95
T2-1. TRMM TSDIS I/F Thread	10d	10/16/95
T2-2. TRMM SDPF I/F Thread	10d	10/16/95
B2 Ingest Build	40d	10/30/95
ETR 2	1d	9/15/95
T3-1. Algorithm I&T Prep. Thread	40d	8/15/95
ETR 4	1d	11/3/95
B3. Algorithm I&T Build	55d	9/15/95
T4-1. ECS Adm. Thread	10d	11/20/95
B4. IR1 Release Build	30d	11/20/95
Consent To Ship Review	1d	12/29/95

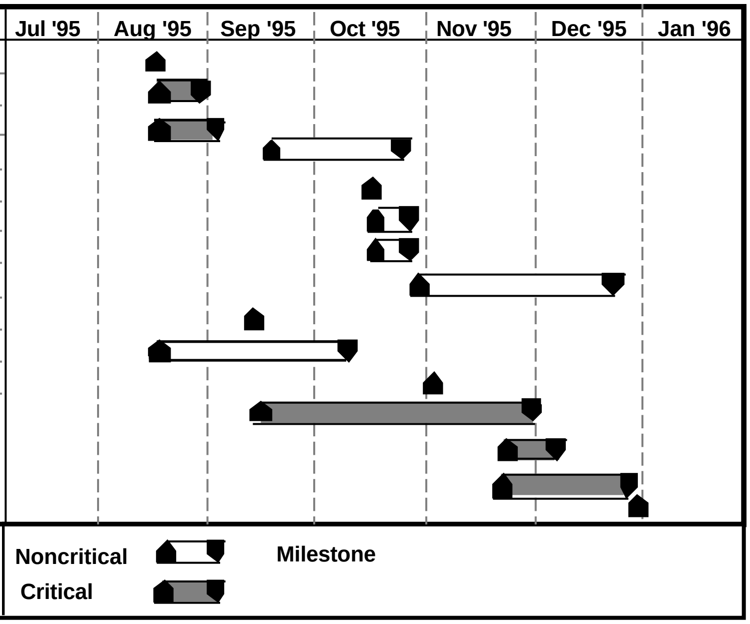

Figure 3-6. Forecasted System I&T IR-1 Schedule

4. Interim Release 1 System Integration and Testing

As described in the Release Plan Content Description white paper (FB9404V2), Interim Release 1 (IR-1) serves two major purposes: 1) early interface support and 2) science software support of TRMM (Tropical Rainfall Measurement Mission), a platform scheduled for launch in August 1997 which relies on ECS to support its mission. To support TRMM data transfer and early interface testing, basic ingest services will be available in IR-1 to interface with TSDIS (TRMM Science Data and Information System) and the GSFC Sensor Data Processing Facility (SDPF). Early interface testing between SCFs (Science Computing Facilities) and ECS interfaces will be available in order to transfer algorithms and algorithm support data.

Deliveries of TRMM CERES (Clouds and Earth's Radiant Energy System) and LIS (Lightning Imaging Sensor) Version 1 algorithms and EOS AM-1 Beta review algorithms are near the end of 1995. PGS (Product Generation System) toolkit deliveries must be made twelve (12) months prior to the Beta reviews for each EOS AM-1 algorithm and twelve (12) months prior to Version 1 delivery for TRMM algorithms.

Figure 4-1 shows the system level builds and threads. 4.x.x refers to the section number within this document specific information about a thread/build will be found.

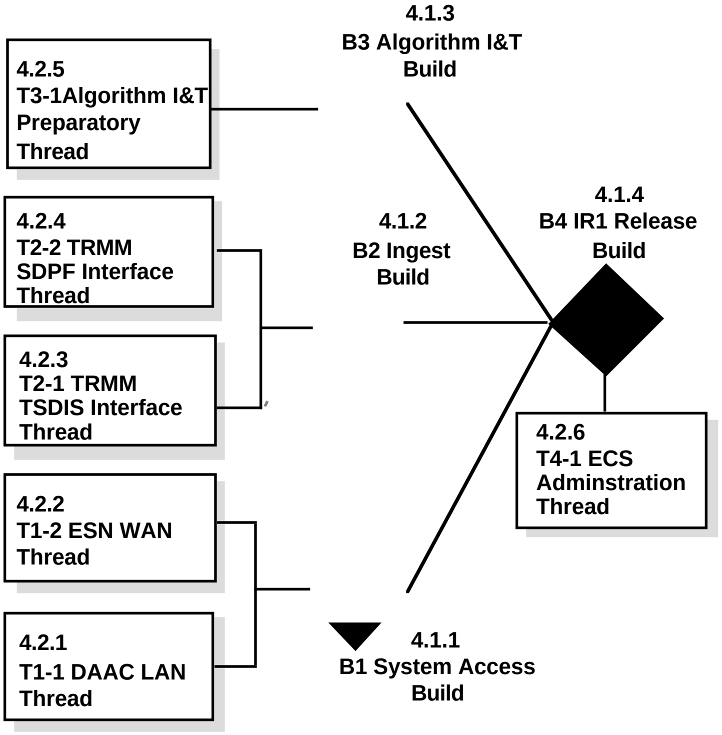

Figure 4-1. Forecasted System Threads/Builds for Interim Release 1

4.1 Interim Release 1 System Builds

This section identifies and describes the functional system level builds which have been defined for Interim Release 1. Each build contains a list of objectives, which describes the overall purpose of the build, and a build test description, which includes a list of dependencies, if applicable, without which the functionality of the build cannot be fully tested. The mapping of the Interim Release 1 requirements to respective system level threads is summarized in Appendix A.

4.1.1 System Access Build

The following listing identifies the test sequences within this build, its paragraph reference number and page number of the accompanying text.

4.1.1.1 Build Objectives

This build provides the capabilities to support Interim Release 1 with a basic suite of communication services required for local and external process-to-process communications. Underlying services include:

- Distributed time and directory services
- File transfer capabilities (ftp, rcp, DFS)
- Remote secure logon/logoff
- LAN and WAN interfaces
- Event logging and reporting

4.1.1.2 Build Test Description

The communication capabilities required for IR-1 are verified beginning with login attempts into the LAN/WAN system. A monitoring account will be used to verify successful logons into the system, while also being used to verify the event logging system. Message passing, via e-mail, and file transfers will be attempted across the LAN and WAN systems to verify network connectivity and basic communications services. Multiple file transfers and message passing will be performed simultaneously to monitor network load and process/CPU access times. Host/network loads and performance will be monitored by the management framework system while multiple accounts will be accessing the system. Finally, the monitoring account will monitor all logoffs for security and verification of complete disconnect. Event logs and reports will be verified through the use of the management framework system and an account used to monitor tester accesses and activities.

Dependencies:

- DAAC LAN Thread
- ESN WAN Thread

Test Support Requirements

- Hardware:
  - Bridges
  - Routers
- Software:
  - System Management Framework
  - Multi-tester emulation capture and playback test tool
- Data:
  - Account names and passwords
  - Filenames for the file transfers
  - Messages to e-mail

4.1.1.2.1 Sequence 1 - Logons

The following series of tests verifies the connectivity to the system (local and remote). Connection is established for testers that enter valid account names and associated passwords. Connection is refused for testers that enter either invalid account names or invalid passwords. In each case, a record of the logon, whether successful or unsuccessful, is recorded in the history log file.

Test Case 1 Local Logons (rlogin) - Valid and Invalid

This test verifies that once connection to the system (Host 1) is established, a tester is able to securely log on to another local host (Host 2), via basic LAN capabilities. All activity for each account is recorded in the history log file.

Input: Valid account names/passwords for accounts A through F (Host 1), “rlogin Host 2” from accounts A through F, valid account names/passwords for accounts A and B (Host 2), invalid account names for accounts C and D (Host 2), valid account names but invalid passwords for accounts E and F (Host 2)

Output: Connection to Host 1 for accounts A through F. Connection to Host 2 for accounts A and B. Messages indicating incorrect logon to Host 2 displayed to accounts C through F. History log file records of all actions made by each account.

Expected Results: Connection to Host 2 is established to accounts A and B, while connection to Host 2 is refused to accounts C through F. Messages indicating an incorrect logon is given to accounts C through F, who remain connected to Host 1. All activity by each account is recorded in the history log file which is verified by the tester.

Test Case 2 Remote Logons (Telnet) - Valid and Invalid

This test verifies that once connection to the system (Host 1), or to any other local host (Host 2), is established, a tester is able to securely log on to a remote host (Host 3), via basic WAN capabilities. All activity for the account is recorded in the history log file.

Input: Valid account names/passwords for accounts A through F (Host 1), “rlogin Host 2” from accounts D through F, valid account names/passwords for accounts D through F (Host 2), “telnet Host 3” from accounts A through F, valid account names/passwords for accounts A and D (Host 3), invalid account names for accounts B and E (Host 3), valid account names but invalid passwords for accounts C and F (Host 3).

Output: Connection to Host 1 for accounts A through F. Connection to Host 2 for accounts D through F. Connection to Host 3 for accounts A and D. Messages to accounts B, C, E, and F indicating incorrect logons and telnet sessions to Host 3 terminated. History log file records of each activity by each account.

Expected Results: Connection to Host 3 is established to accounts A and D, while connection to Host 3 is refused to accounts B, C, E, and F. Messages indicating an incorrect logon is given to accounts B, C, E, and F. Accounts B and C remain connected to Host 1 and accounts E and F remain connected to Host 2. All activity by each account is recorded in the history log file which is verified by the tester.

Test Case 3 Remote Logons (Telnet) - Valid and Invalid

This test verifies that once a connection to a remote host (Host 3) from the local host (Host 1), a tester is able to securely log back into the local host (Host 1), via basic WAN capabilities. This test verifies that connection into the LAN from the WAN is possible. All activity for the account is recorded in the history log file.

Input: Valid account names/passwords for accounts A through F (Host 1), "telnet Host 3" from accounts A through F, valid account names/passwords for Host 3. "telnet Host 1" from accounts A through F from Host 3, valid account names/passwords for accounts A and D (Host 1), invalid account names for accounts B and E (Host 1), valid account names but invalid passwords for accounts C and F (Host 1).

Output: Connection from Host 1 for accounts A through F. Connection to Host 3 for accounts A through F. Connection to Host 1 for accounts A and D. Messages to accounts B, C, E, and F indicating incorrect logons and telnet sessions to Host 1 terminated. History log file records of each activity by each account.

Expected Results: Connection to Host 1 is established for accounts A and D, while connection to Host 1 is refused for accounts B, C, E, and F. All activity by each account is recorded in the history log file which is verified by the tester.

Test Case 4 Verification of History Log File

This test verifies that the history log file records all system accesses and activities. Manually, the tester will review the history log file to verify that tests 1-6 were completely recorded. The history log should contain correct information for the activities which occurred in this test sequence.

Input: Test cases 1-3 of this sequence.

Output: History log file print out.

Expected Results: The history log should reflect all activities which occurred during this test sequence.

4.1.1.2.2 Sequence 2 - Logoffs

The following test verifies that connection to the system, either locally or remotely, is properly closed once a logoff to the system, either normally or abnormally, is encountered.

Test Case 1 Logoffs - Normal

This test verifies that when a tester, using a valid account, logs off a system or a host, the connection is properly closed to the system or the host. In this case, account A connects to the system on Host1, logs on to a local host (Host 2), then logs on to a remote host (Host 3). Subsequently, the tester logs off the remote host (Host 3), then the local host (Host 2), and finally Host 1. In each instance, another tester, using another valid account, is monitoring the system on each host to verify that connection to the host has been closed for account A. Also, all logon and logoff activity is recorded in the history log file.

Input: Valid account names/passwords for accounts A and B on Hosts 1, 2, and 3, “rlogin Host 2”, “telnet Host 3”, logoff.

Output: Account B monitors the activity of account A, history log file records that have corresponding logon and logoff records.

Expected Results: Account B verifies that connection is established for account A on the different hosts when account A logs on and that connection is closed when account A is logged off each host. History log files should have corresponding logon and logoff records.

Test Case 2 Logoffs - Abnormal

This test verifies that when an abnormal event occurs and disconnects an account from the system (i.e., the tester's workstation is turned off, one of the hosts is powered off, a UNIX "kill", etc.), the system properly closes connection to the account. History log file entries are recorded for all activity. This test is to be repeated for the different types of abnormal events that will cause a disconnect from the system.

Input: Valid account names/passwords for accounts A and B on Hosts 1, 2, and 3, "rlogin Host 2", "telnet Host 3", an abnormal termination of account A to the system.

Output: Account B monitors activity of account A, history log file records which include a record that indicates that account A was logged out of Host X due to a system error. Port connection the host is properly closed.

Expected Results: History log file records which include a record that indicates that account A was logged out of Host X due to a system error. The port connection to the host is closed. If one of the hosts was powered off, then the tester waits to log back on once the host is powered back on.

Test Case 3 Verification of History Log File

This test verifies that the history log file records all system accesses and activities. Manually, the tester will review the history log file to verify that tests 1-2 were completely recorded. The history log should contain correct information for the activities which occurred in this test sequence.

Input: Test cases 1-2 of this sequence.

Output: History log file print out.

Expected Results: The history log should reflect all activities which occurred during this test sequence.

4.1.1.2.3 Sequence 3 - Access and Privilege Control

The following tests verify the control of accesses and privileges for accounts, devices, and for the Directory Service. The control of accesses and privileges comes from the security registry which is based on authentication. The security registry is maintained by an administrator who assigns an account to a group name. The administrator places privileges on the group name so that any account that is assigned to that particular group name adopts the same privileges.

Test Case 1 Accessing Devices and Peripherals Defined to a Group

This test verifies that an account that is registered within a group is able to access the different devices and peripherals that have been assigned to the particular group.

Input: Valid account names/password for account A (account is assigned to Group A). Attempt access to devices and peripherals specific to Group A.

Output: Access allowed to each device and peripheral.

Expected Results: Access is allowed to each device and peripheral specific to Group A.

Test Case 2 Accessing Devices and Peripherals Not Defined to a Group

This test verifies that an account that is registered within a group is unable to access the different devices and peripherals that have not been assigned to the particular group.

Input: Valid account names/password for account A (account is assigned to Group A). Attempt access to devices and peripherals not specific to Group A.

Output: Access disallowed to each device and peripheral.

Expected Results: Access is disallowed to each device and peripheral not specific to Group A.

Test Case 3 Directory Service (privileges allowed)

This test verifies that an account that is registered within a group is able to use the privileges defined when using the Directory Service.

Input: Valid account name/password for account A (account is assigned to Group A). Attempt to use privileges allowed (file access, directory update, create/delete a directory, etc.) by Group A.

Output: Each privilege is allowed by account A.

Expected Results: Account A is allowed to use all the privileges allocated to Group A.

Test Case 4 Directory Service (privileges not allowed)

This test verifies that an account that is registered within a group is not able to use the privileges not defined when using the Directory Service.

Input: Valid account name/password for account A (account is assigned to Group A). Attempt to use privileges not allowed (file access, directory update, create/delete a directory, etc.) by Group A.

Output: Each privilege is disallowed by account A.

Expected Results: Account A is not allowed to use the privileges not allocated to Group A.

4.1.1.2.4 Process to Process Communication (RPC Calls)

The following test verifies the ability for clients and servers to communicate with each other. Furthermore, the client can be located on the same host or on another host (each case will be tested).

Test Case 1 Client Requires Response from the Server

This test verifies the ability of a client to bind with a server. The client will then perform some operation based on the response received from the server. Develop a client/server application to pass a set of parameters between the client and the server. Initialize the server and verify that the server is up and waiting to be called by a client. Run the client to pass a set of parameters back from the server and verify that the set of numbers was passed between the client and server.

Input: Transfer of data between client and server and, between server and client.

Output: Screen output showing that the server is up and running. Screen output to show that the client passed the data. History log indicating that the server was initialized and the client was started.

Expected Results: Screen output showing that the server is up and running. Screen output to show that the client passed the data. History log indicating that the server was initialized and the client was started.

Test Case 2 Client Does Not Require Response from the Server

This test verifies the ability of a client to bind with a server. The client does not require a response from the server. An example would be a user survey where the client would retrieve the questions from the server. Develop a client/server application in which the server will perform a transaction when initialized by the client. Run the server (in the background) and verify that the server is up and waiting to be called by the client. Initialize the client and verify the user survey application.

Input: Initialize server. Communication between client and server. Initialize transaction.

Output: Screen output showing that the server is up and running. Screen outputs showing the results of the transaction. The client accessing of the user survey questions. History log indicating that the server was initialized and the client was started.

Expected Results: Screen output showing that the server is up and running. Screen outputs showing the results of the transaction. The client accessing of the user survey questions. History log indicating that the server was initialized and the client was started.

4.1.1.2.5 Sequence 5 - Communication via Internet

The following series of tests verify that testers (by e-mail) are able to communicate to local and remote systems, via basic LAN and WAN capabilities, and that all activity is recorded in the history log file.

Test Case 1 Sending E-Mail Messages to Local and Remote Hosts

This test verifies that a tester using a valid account, via basic LAN and WAN capabilities, is able to e-mail messages to other accounts on the same host (Host 1), on a local host (Host 2), or on a remote (Host 3) host. In all cases, records of the transactions are recorded in the history log file.

Input: Valid account names/passwords for account A on local Host 1, account B on local Host 1, account C on local Host 2, and account D on remote Host 3. Message 1 sent from account A to account B. Message 2 sent from account A to account C. Message 3 sent from account A to account D.

Output: Connection to the respective hosts, message 1 received by account B, message 2 received by account C, and message 3 received by account D. History log file records of all activity and transactions by the tester.

Expected Results: The history log will record the logon to the system by each account, it will record the transmission of the e-mail messages, it will record the resource usage, response time, and the number of transactions.

Test Case 2 Receiving E-mail Messages from Local and Remote Hosts

This test verifies that a tester using a valid account, via basic LAN and WAN capabilities, is able to receive e-mail messages from the same host (Host 1), a local host (Host 2), or a remote host (Host 3). Each message, in this case, will be sent simultaneously to the same account (account A). In all cases, records of the transactions are recorded in the history log file.

Input: Valid account names/passwords for account A on local Host 1, account B on local Host 1, account C on local Host 2, and account D on remote Host 3. Messages from accounts B, C, and D to account A.

Output: Connection to the respective hosts, messages received by account A. History log file records of all activity and transactions by the tester.

Expected Results: The history log will record the logon to the system by each account, it will record the transmission of the e-mail messages, it will record the resource usage, response time, and the number of transactions.

Test Case 3 Verification of History Log File

This test verifies that the history log file records all system accesses and activities. Manually, the tester will review the history log file to verify that tests 1-2 were completely recorded. The history log should contain correct information for the activities which occurred in this test sequence.

Input: Test cases 1-2 of this sequence.

Output: History log file print out.

Expected Results: The history log should reflect all activities which occurred during this test sequence.

4.1.1.2.6 Sequence 6 - Transfer Data Files

The following series of tests demonstrates that a valid account with the proper system access is able to transfer data files of various sizes from one machine to another.

Test Case 1 FTP Data File Locally

This test verifies that a tester, using a valid account, is able to transmit data files of various sizes to an account on another local machine via ftp through the Ethernet LAN.

Input: Valid login for account A on Host 1 and Host 2. Data files for file transfer (ftp) from Host 1 to Host 2.

Output: ftp completes data file transfers from Host 1 to Host 2. Host 2 directory listings verify that the data files were transferred.

Expected Results: The history log will record the logon onto the systems by account A, it will record the transmission of the FTP transactions, it will record the resource usage, response time, and the number of transactions. Checksum of data files on Host 1 and Host 2 should be equal.

Test Case 2 FTP Data File within WAN

This test verifies that a tester, using a valid account, is able to transmit data files of various sizes to a remote machine via ftp through the WAN.

Input: Valid login for tester on Host 1 at EDF and Host 2 at LaRC. Data files for transfer (ftp) from Host 1 to Host 2.

Output: ftp completes data file transfers from Host 1 to Host 2. Host 2 directory listings will verify that the data files were transferred.

Expected Results: The history log will record the logon to the machines, it will record the transmission of the ftp transactions, it will record the resource usage, response time, and the number of transactions. Checksum of data files on Host 1 and Host 2 should be equal.

Test Case 3 RCP Data File within LAN

This test verifies that a tester, using a valid account, is able to transmit data files of various sizes to an account on another machine via remote file copy (rcp).

Input: Valid login for account A on Host 1 and Host 2. Data files for remote copy (rcp) from Host 1 to Host 2.

Output: rcp completes data file transfers from Host 1 to Host 2. Host 2 directory listings verify that the data files were transferred.

Expected Results: The history log will record the logon onto the systems by account A, it will record the transmission of the rcp transactions, it will record the resource usage, response time, and the number of transactions. Checksum of data files on Host 1 and Host 2 should be equal.

Test Case 4 RCP Data File Through WAN

This test verifies that a tester, using a valid account, is able to transfer data files of various sizes to a remote machine via rcp through the WAN.

Input: Valid login for account on Host 1 at EDF and Host 2 at a remote site supported in IR-1. Data files for remote copy (rcp) from Host 1 to Host 2.

Output: rcp completes data file transfers from Host 1 to Host 2. Host 2 directory listings verify that the data files were transferred.

Expected Results: The history log will record the logon onto the systems, it will record the transmission of the rcp transactions, it will record the resource usage, response time, and the number of transactions. Checksum of data files on Host 1 and Host 2 should be equal.

Test Case 5 Transfer of File Using RPC Pipes

This test verifies that a tester, using a valid account, is able to transfer data files of various sizes using RPC Pipes.

Input: Valid login for account on Host 1 at EDF and Host 2 at a remote site supported in IR-1. Data files for file transfer using RPC Pipes.

Output: Completed data file transfers from Host 1 to Host 2. Host 2 directory listings verify that the data files were transferred.

Expected Results: The history log will record the logon onto the systems, it will record the transmission of the RPC Pipe transactions, it will record the resource usage, response time, and the number of transactions. Checksum of data files on Host 1 and Host 2 should be equal.

Test Case 6 Transfer of File through Distributed File Service

This test verifies that testers, using valid accounts with appropriate authenticity and authorization, are able to access data files of various sizes from any other host (within LAN or WAN) via DCE Distributed File Service. When given a filename, a DFS client cache manager queries the Cell Directory Service (CDS) for the address of a file set location server. The cache manager stores the address for subsequent use. The cache manager makes a remote procedure call to the file set location server to get the address of the file server serving the target file. The cache manager makes a remote procedure call (request to open file) to the file server. The file server verifies the client's authenticity and, if the client is authorized, serves the file to the client.

Input: Valid login for accounts A, B, C, and D on local Host 1, valid login for accounts B and D on remote Host 2. Same filename request to Distributed File Service by all accounts. Only accounts A and B have appropriate authorization to transfer file.

Output: Accounts A and B receive data file requested on the respective host. Host directory listings verify that the data file was transferred. Message indicating lack of appropriate authorization to transfer file received by accounts C and D.

Expected Results: File is transferred into the local listings of the respective host for account's A and B. A comparison of the local copy of the file to that of the remote copy should produce the same results. Accounts C and D receive message that the accounts did not have the appropriate authorization. The history log will record the logons and the file transmissions.

Test Case 7 Verification of History Log File

This test verifies that the history log file records all system accesses and activities. Manually, the tester will review the history log file to verify that tests 1-6 were completely recorded. The history log should contain correct information for the activities which occurred in this test sequence.

Input: Test cases 1-6 of this sequence.

Output: History log file print out.

Expected Results: The history log should reflect all activities which occurred during this test sequence.

4.1.1.2.7 Sequence 7 - Multiple Transaction Requests

The following series of tests demonstrate the systems ability to withstand multiple transaction requests from multiple accounts over the LANs and WANs.

Test Case 1 Multiple Accounts Transmitting Large Data Files to GSFC DAAC

This test verifies that multiple accounts are able to transmit large data files over the WAN to GSFC.

Input: Accounts A through F are logged on with valid account names/passwords. Accounts A through F will simultaneously transmit a large data file to GSFC, using ftp the "hash" option, to valid accounts at GSFC.

Output: Accounts A through F will view their perspective screens and notice a slight delay in the rate the hashing appears.

Expected Results: The history log will record each input that was initiated by each account. The system response will slow down due to the network traffic from the multiple ftp transactions

Test Case 2 Multiple Accounts Transmitting Large Data Files within the EDF

This test verifies that multiple accounts can transmit large data files over the Ethernet LAN within the EDF.

Input: Accounts A through G are logged on with valid account names/passwords. Accounts A through F will simultaneously transmit a large data file within the EDF to account G using ftp with the "hash" option.

Output: Accounts A through F will view their perspective screens and notice a slight delay in the rate the hashing appears.

Expected Results: The history log will record each input that was initiated by each account. The system response will slow down due to the network traffic from the multiple ftp transactions.

Test Case 3 Verification of History Log File

This test verifies that the history log file records all system accesses and activities. Manually, the tester will review the history log file to verify that tests 1-2 were completely recorded. The history log should contain correct information for the activities which occurred in this test sequence.

Input: Test cases 1-2 of this sequence.

Output: History log file print out.

Expected Results: The history log should reflect all activities which occurred during this test sequence.

4.1.1.2.8 Sequence 8 - Accessing Hosts on LAN/WAN

The following series of tests verify that all host machines within the local area network and the wide area network are all accessible from any other host machine in the same system. These series of tests will be verified at all sites.

Test Case 1 Accessing Hosts within LAN

This test verifies that all host machines connected within the LAN are accessible through the network from other hosts in the same LAN. This test will be verified at all sites.

Input: Valid login for tester on Host 1. "Ping" all host machines that on LAN.

Output: ping statistics displayed for each machine pinged.

Expected Results: All host machines should return ping messages that are connected to LAN. History log will record all activities.

Test Case 2 Accessing Hosts within WAN

This test verifies that all host machines connected within the WAN are accessible through the network from other hosts within the WAN. This test will verify network connection across all sites.

Input: Valid login for tester on Host 1. "Ping all host machines connected within WAN, including LAN.

Output: ping statistics displayed for each machine pinged.

Expected Results: All host machines should return ping messages that are connected within WAN/LAN. History log will record all activities.

Test Case 3 Verification of History Log File

This test verifies that the history log file records all system accesses and activities. Manually, the tester will review the history log file to verify that tests 1-2 were completely recorded. The history log should contain correct information for the activities which occurred in this test sequence.

Input: Test cases 1-2 of this sequence.

Output: History log file print out.

Expected Results: The history log should reflect all activities which occurred during this test sequence.

4.1.1.2.9 Sequence 9 - V0 Network Access

The following series of tests verify the V0 network connectivity used in IR-1. All tests will be performed on host machines within each site (GSFC, MSFC, LaRC, and EDF) that are directly connected to the V0 network. Since the V0 network is a dedicated network for data transfer only, the tester will "ping" dedicated host machines at the other sites from the dedicated host machine at GSFC.

Test Case 1 GSFC to EDF V0 Access

This test verifies that the dedicated V0 link from GSFC to EDF is operational. To perform this test the tester must logon to the dedicated V0 host machine within GSFC to test the "dedicated V0 link to EDF".

Input: Valid name/password for tester on dedicated V0 host machine at GSFC. 'ping' dedicated V0 host machine at EDF.

Output: Successful logon to V0 host machine. ping statistics from V0 host machine at EDF.

Expected Results: V0 host machine at EDF should return ping messages. History log file will record all test activities and ping statistics.

Test Case 2 GSFC to LaRC

This test verifies that the dedicated V0 link from GSFC to LaRC is operational. To perform this test the tester must logon to the dedicated V0 host machine within GSFC to test the "dedicated V0 link to LaRC".

Input: Valid name/password for tester on dedicated V0 host machine at GSFC. 'ping' dedicated V0 host machine at LaRC.

Output: Successful logon to V0 host machine. ping statistics from V0 host machine at LaRC.

Expected Results: V0 host machine at LaRC should return ping messages. History log file will record all test activities and ping statistics.

Test Case 3 GSFC to MSFC

This test verifies that the dedicated V0 link from GSFC to MSFC is operational. To perform this test the tester must logon to the dedicated V0 host machine within GSFC to test the V0 link to MSFC.

Input: Valid name/password for tester on dedicated V0 host machine at GSFC. 'ping' dedicated V0 host machine at GSFC.

Output: Successful logon to V0 host machine. ping statistics from V0 host machine at MSFC.

Expected Results: V0 host machine at MSFC should return ping messages. History log file will record all test activities and ping statistics.

Test Case 4 Verification of History Log File

This test verifies that the history log file records all system accesses and activities. Manually, the tester will review the history log file to verify that tests 1-3 were completely recorded. The history log should contain correct information for the activities which occurred in this test sequence.

Input: Test cases 1-3 of this sequence.

Output: History log file print out.

Expected Results: The history log should reflect all activities which occurred during this test sequence.

4.1.1.2.10 Sequence 10 - NSI Interface

The following test verifies connectivity to NSI.

Test Case 1 NSI Connectivity

Input: Valid account name/password on a V0 client.

Output: Communication to an NSI client.

Expected Results: NSI interface is confirmed.

4.1.2 Ingest Build

The following listing identifies the test sequences within this build, its paragraph reference number and page number of the accompanying text.

4.1.2.1 Build Objectives

The objective of this build verification test is to evaluate the ingest interface functions for the Interim Release 1 (IR-1) system release by ingesting data identified for IR-1. This includes:

- receiving and acknowledging the receipt of the Data Availability Notice (DAN)
- receiving, validating and acknowledging the ingest of the data; logging the receipt
- validation and ingest of the data; and initiating retransmission of the DAN or data, if necessary, due to a transmission or validation error.

This build will demonstrate the capability to transfer data for ingest over a Local Area Network (LAN) or Wide Area Network (WAN) or to ingest data via physical electronic media. There is no data archiving or data processing required for IR-1.

4.1.2.2 Build Test Description

For IR-1, the ECS DAACs will interface with the Tropical Rainfall Measuring Mission (TRMM) Science Data and Information System (TSDIS), the Sensor Data Processing Facility (SDPF) located at the Goddard Space Flight Center (GSFC), and the National Oceanic and Atmospheric Administration's (NOAA) National Environmental Satellite Data, and Information Service (NESDIS) to ingest the following simulated data: TRMM L0, processed TRMM, metadata, ancillary and engineering, in Hierarchical Data Format (HDF), Common Data Format (CDF) or native format.

To support the ingest interface testing, basic ingest services will be available at the GSFC Distributed Active Archive Center (DAAC), the Marshall Space Flight Center (MSFC) DAAC, and the Langley Research Center (LaRC) DAAC. Interfaces not fully developed will be simulated.

DANs are sent from the TSDIS, SDPF, and NESDIS to an ECS DAAC. The ECS DAACs receive the DANs and send a DAN receipt acknowledgment to the sender of the DAN. The data is then transmitted in either a data driven or schedule driven mode to the ECS DAACs. The transmission is monitored for errors. After the data is successfully received it is validated by comparing it to the DAN as well as checking the header of the L0 data. If there was an error in the transmission or in the validation process a retransmission, up to a predetermined number of times, is initiated. When the data is successfully validated it is then staged and a successful data ingest acknowledgment is sent to the data provider. Ingest management includes the use of the System Management Framework Utility for monitoring of the hardware and software used for the ingest process as well as the File Directory Management system for use in managing the ingested data.

Dependencies: (If Applicable)

- System Management Framework Utility

Test Support Requirements

- Hardware:

  - Sending host workstation
  - Receiving host workstation
  - Gateway/Router to WAN
  - LAN hardware
  - WAN hardware
  - Workstation for ECS Management/Monitor Utility

- Software:

  - Ingest application
  - Communications
  - Simulated interfaces (as needed)
  - System Management Framework Utility

- Data: (HDF, CDF, native format)

  - Simulated TRMM Level 0 data and metadata
  - Simulated TRMM data products and metadata
  - Ancillary data
  - Engineering data

4.1.2.2.1 Sequence 1 - Receipt and Validation of Ingest Data

The following series of tests verify that data, of all formats and types identified for IR-1, transmitted for ingest can been received and validated successfully. These tests also verify that retransmissions of the data, up to a predetermined number of times, are initiated in response to an error in the transmission of the data via LAN or WAN or an error in the validation of the data. Receipt includes monitoring of data transfer via LAN or WAN for transmission errors. Validation includes identifying that the data received is the same as what was intended to be transferred, identifying the data format and checking the header information.

<u>Test Case 1</u> <u>Successful Receipt and Validation</u>

This test verifies that data transferred for ingest can been successfully received and validated.

Input: DAN with corresponding valid data, in all valid formats and types, including the following data types transferred electronically to the ECS:

Valid - TRMM L0, TRMM products, metadata, ancillary data and engineering data.

Output: Data receipt logs, validation logs and directory listings.

Expected Results: All data transferred for ingest has been successfully received and validated by the receiving ECS DAAC. The data receipt and validation logs contain entries for all transmitted data. The directory listings contain files for all of the data transferred.

Test Case 2 Unsuccessful Receipt and Validation

This test verifies that when errors are encountered with the receipt or validation of the data transferred for ingest, retransmissions of the data, up to a predetermined number of times, are initiated. Also, if the ingest of the data is still unsuccessful after the predetermined number of tries a message is sent to the sender indicating the error(s).

Input: DAN with corresponding mismatched data, DAN with corresponding data containing errors in the headers.

Output: Error logs, retransmission attempts, ingest error messages and directory listings.

Expected Results: The data transferred for ingest has been unsuccessfully received or validated by the receiving ECS DAAC. The error logs contain entries for all transmission and validation errors. Retransmissions of the data are initiated up to the predetermined number of times. The ingest error messages are received by the senders. The directory listings contain no files for the data unsuccessfully received or validated.

4.1.2.2.2 Sequence 2 - Data Ingest Utilizing Data Availability Notices

The following series of tests verify ingest of all data formats, types, and platforms identified for IR-1, utilizing basic communications, based on a previously transmitted DAN. DANs are provided by the sender as an input to data ingest. These DANs contain information regarding data identification, data format, data granule size, data location and estimated time of availability or time of transmission. The DAN for "Data-Driven" ingest is used to inform the ECS DAAC of the planned transfer of data for ingest. The DAN for "Schedule-Driven" ingest will notify the ECS DAAC as to the period for which the data remains available for ingest. DANs are transferred via a supporting network to an ECS DAAC. The ECS DAAC will respond by transmitting a receipt acknowledgment of the DAN to the sender. Either, the sender will initiate transmission to the ECS DAAC or the ECS DAAC will initiate the transmission, of the data identified in the DAN depending on whether it is a data driven "push" or a schedule driven "pull" type of transfer. The data is then validated and put of disk for staging and a message acknowledging successful receipt of the data is transmitted to the sender.

Test Case 1 DAN Receipt and Acknowledgment

This test verifies that a DAN can be successfully transferred, using basic communications, to an ECS DAAC (for all identified IR-1 platforms) and that an acknowledgment of the DAN receipt is transmitted to the sender .

Input: A DAN for transmission to an ECS DAAC. Input will include transfer of the DAN to all platforms identified for IR-1, using all ECS protocols.

Output: Receipt logs and DAN receipt acknowledgment messages.

Expected Results: The DANs are successful received by the ECS DAAC and the DAN receipt acknowledgment messages are received by the senders. The receipt logs contain entries for the DANs.

Test Case 2 Data Driven Ingest

This test verifies successful data driven (push) ingest for all data formats, types, and platforms identified for IR-1, utilizing basic communications, based on a previously transmitted DAN. A copy of all data to be ingested is made prior to data transmission. This will act as a baseline so comparisons can be made of the data prior to and after transmission.

Input: A DAN containing at least one data set for each data format and type and the corresponding data for transmission to an ECS DAAC. Inputs will include transfer of the DAN and the corresponding data to all platforms identified for IR-1, using all ECS protocols.

Output: Receipt logs, DAN and data receipt acknowledgment messages, directory listings and data comparisons.

Expected Results: All DANs and their corresponding data sets are successfully received, validated and staged to disk. The receipt logs contain entries for all DANs and their corresponding data sets. The DAN and data receipt acknowledgment messages are received by the sender. The directory listings contain files for all of the data sets transferred. The comparisons of the data prior to and after transmission show no differences.

Test Case 3 Schedule Driven Ingest for Valid Data

This test verifies successful schedule driven (pull) ingest for all data formats, types, and platforms identified for IR-1, utilizing basic communications, based on a previously transmitted DAN. A copy of all data to be ingested is made prior to data transmission. This will act as a baseline so comparisons can be made of the data prior to and after transmission.

Input: A DAN containing at least one data set for each data format and type and the corresponding data for transmission to an ECS DAAC. Inputs will include transfer of the DAN and the corresponding data to all platforms identified for IR-1, using all ECS protocols.

Output: Receipt logs, DAN and data receipt acknowledgment messages, directory listings and data comparisons.

Expected Results: All DANs and their corresponding data sets are successfully received, validated and staged to disk. The receipt logs contain entries for all DANs and their corresponding data sets. The DAN and data receipt acknowledgment messages are received by the sender. The directory listings contain files for all of the data sets transferred. The comparisons of the data prior to and after transmission show no differences.

4.1.2.2.3 Sequence 3 - Data Ingest Without Previously Transmitted DAN

The following test verifies the system reaction to a receipt of data without previously receiving a corresponding DAN.

<u>Test Case 1</u> <u>Ingest for Valid Data</u>

This test verifies the unsuccessful ingest of data transferred without previously receiving a corresponding DAN.

Input: Data for transmission to an ECS DAAC.

Output: Error log, ingest error messages and directory listings.

Expected Results: The data is unsuccessfully ingested. The error log contains entries for the transferred data. The ingest error messages are received by the sender. The directory listings do not contain file entries for the transmitted data.

4.1.2.2.4 Sequence 4 - Data Ingest via the Local Area Network (LAN)

The following series of tests verify ingest of data identified for IR-1, utilizing LAN communications, based on a previously transmitted DAN.

<u>Test Case 1</u> <u>DAN transfer via LAN</u>

This test verifies that a DAN can be successfully transferred, using LAN communications, to an ECS DAAC and that an acknowledgment of the receipt is transmitted to the sender.

Input: A DAN for transmission to an ECS DAAC.

Output: Receipt log and DAN receipt acknowledgment message.

Expected Results: The DAN is successful received by the ECS DAAC and the DAN receipt acknowledgment message is received by the sender. The receipt log contain an entry for the DAN.

Test Case 2 Data Driven Ingest via LAN

This test verifies successful data driven (push) ingest for data identified for IR-1, utilizing LAN communications, based on a previously transmitted DAN. A copy of all data to be ingested is made prior to data transmission. This will act as a baseline so comparisons can be made of the data prior to and after transmission.

Input: A DAN and the corresponding data for transmission to an ECS DAAC.

Output: Receipt log, DAN and data receipt acknowledgment messages, directory listings and data comparisons.

Expected Results: The DAN and the corresponding data sets are successfully received, validated and staged to disk. The receipt log contains an entry for the DAN and the corresponding data sets. The DAN and data receipt acknowledgment messages are received by the sender. The directory listings contain files for all of the data sets transferred. The comparisons of the data prior to and after transmission show no differences.

Test Case 3 Schedule Driven Ingest for via LAN

This test verifies successful schedule driven (pull) ingest for all data identified for IR-1, utilizing LAN communications, based on a previously transmitted DAN. A copy of all data to be ingested is made prior to data transmission. This will act as a baseline so comparisons can be made of the data prior to and after transmission.

Input: A DAN and the corresponding data for transmission to an ECS DAAC.

Output: Receipt log, DAN and data receipt acknowledgment messages, directory listings and data comparisons.

Expected Results: The DAN and the corresponding data sets are successfully received, validated and staged to disk. The receipt log contains an entry for the DAN and the corresponding data sets. The DAN and data receipt acknowledgment messages are received by the sender. The directory listings contain files for all of the data sets transferred. The comparisons of the data prior to and after transmission show no differences.

4.1.2.2.5 Sequence 5 - Data Ingest via Wide Area Network (WAN)

The following series of tests verify ingest of data identified for IR-1, utilizing WAN communications, based on a previously transmitted DAN.

Test Case 1 DAN Transfer via WAN

This test verifies that a DAN can be successfully transferred, using WAN communications, to an ECS DAAC and that an acknowledgment of the receipt is transmitted to the sender.

Input: A DAN for transmission to an ECS DAAC.

Output: Receipt log and DAN receipt acknowledgment message.

Expected Results: The DAN is successful received by the ECS DAAC and the DAN receipt acknowledgment message is received by the sender. The receipt log contain an entry for the DAN.

Test Case 2 Data Driven Ingest via WAN

This test verifies successful data driven (push) ingest for data identified for IR-1, utilizing WAN communications, based on a previously transmitted DAN. A copy of all data to be ingested is made prior to data transmission. This will act as a baseline so comparisons can be made of the data prior to and after transmission.

Input: A DAN and the corresponding data for transmission to an ECS DAAC.

Output: Receipt log, DAN and data receipt acknowledgment messages, directory listings and data comparisons.

Expected Results: The DAN and the corresponding data sets are successfully received, validated and staged to disk. The receipt log contains an entry for the DAN and the corresponding data sets. The DAN and data receipt acknowledgment messages are received by the sender. The directory listings contain files for all of the data sets transferred. The comparisons of the data prior to and after transmission show no differences.

Test Case 3 Schedule Driven Ingest for via WAN

This test verifies successful schedule driven (pull) ingest for all data identified for IR-1, utilizing WAN communications, based on a previously transmitted DAN. A copy of all data to be ingested is made prior to data transmission. This will act as a baseline so comparisons can be made of the data prior to and after transmission.

Input: A DAN and the corresponding data for transmission to an ECS DAAC.

Output: Receipt log, DAN and data receipt acknowledgment messages, directory listings and data comparisons.

Expected Results: The DAN and the corresponding data sets are successfully received, validated and staged to disk. The receipt log contains an entry for the DAN and the corresponding data sets. The DAN and data receipt acknowledgment messages are received by the sender. The directory listings contain files for all of the data sets transferred. The comparisons of the data prior to and after transmission show no differences.

4.1.2.2.6 Sequence 6 - Data Ingest Faults

The following series of tests evaluate system reaction to LAN/WAN hardware faults and ingest application faults during the data ingest process. The system management framework utility will be monitored for hardware and application process alerts related to data ingest. Hardware faults will be caused by simulated power loss or disconnect from LAN/WAN. Ingest application faults will be simulated by terminating processes. These tests will also verify that after the fault has been corrected the ingest process can be resumed at the stage at which the fault occurred.

Test Case 1 Gateway/Router Fault

This test verifies that the system management framework utility can isolate, locate, identify an characterize the LAN/WAN gateway/router fault, properly log all events and that the ingest process can be resumed at the stage at which the fault occurred. During the test, data ingest must be in progress while the gateway/router fault occurs. The Gateway/Router fault will be made to occur during four stages of the ingest process. During DAN transmission, before DAN receipt acknowledgment, during data transmission and before data receipt acknowledgment.

Input: A DAN and the corresponding data for transmission to an ECS DAAC. Simulated gateway/router power loss faults. Correction of simulated gateway/router power loss faults.

Output: System management framework display and logs. Ingest process continuation at stage of fault occurrence.

Expected Results: System management framework utility isolates, locates, identifies and characterizes the gateway/router faults and logs the faults. Ingest process resumes, after faults are corrected, at the stage at which the faults occurred.

Test Case 2 Sender Host Fault

This test verifies that the system management framework utility can isolate, locate, identify and characterize the senders host machine fault, properly log all events and that the ingest process can be resumed at the stage at which the fault occurred. During the test, data ingest must be in progress while the senders host machine fault occurs. The sender host machine fault will be made to occur during four stages of the ingest process. During DAN transmission, before DAN receipt acknowledgment, during data transmission and before data receipt acknowledgment.

Input: A DAN and the corresponding data for transmission to an ECS DAAC. Simulated sender host machine power loss faults. Correction of simulated sender host machine power loss faults.

Output: System management framework display and logs. Ingest process continuation at stage of fault occurrence.

Expected Results: System management framework utility isolates, locates, identifies and characterizes the sender host machine faults and logs the faults. Ingest process resumes, after faults are corrected, at the stage at which the faults occurred.

Test Case 3 Receiving Host Fault

This test verifies that the system management framework utility can isolate, locate, identify and characterize the receiving host machine fault, properly log all events and that the ingest process can be resumed at the stage at which the fault occurred. During the test, data ingest must be in progress while the receiving host machine fault occurs. The receiving host machine fault will be made to occur during four stages of the ingest process. During DAN transmission, before DAN receipt acknowledgment, during data transmission and before data receipt acknowledgment.

Input: A DAN and the corresponding data for transmission to an ECS DAAC. Simulated receiving host machine power loss faults. Correction of simulated receiving host machine power loss faults.

Output: System management framework utility display and logs. Ingest process continuation at stage of fault occurrence.

Expected Results: System management framework utility isolates, locates, identifies and characterizes the receiving host machine faults and logs the faults. Ingest process resumes, after faults are corrected, at the stage at which the faults occurred.

Test Case 4 Ingest Application Fault

This test verifies that the system management framework utility can isolate, locate, identify and characterize the ingest application fault, properly log all events and that the ingest process can be resumed at the stage at which the fault occurred. During the test, data ingest must be in progress while the ingest application fault occurs. The receiving host machine fault will be made to occur during four stages of the ingest process. During DAN transmission, before DAN receipt acknowledgment, during data transmission and before data receipt acknowledgment.

Input: A DAN and the corresponding data for transmission to an ECS DAAC. Ingest application (terminate process) faults. Correction of ingest application faults.

Output: System management framework utility display and logs. Ingest process continuation at stage of fault occurrence.

Expected Results: System management framework utility isolates, locates, identifies and characterizes the ingest application faults and logs the faults. Ingest process resumes, after faults are corrected, at the stage at which the faults occurred.

4.1.2.2.7 Sequence 7 - Ingest Physical Electronic Media

The following series of tests verify the system's ability to successfully ingest data from physical electronic media types identified for IR-1 and to request a new copy of the media if the original media is corrupted. For IR-1, the following media will be used to ingest data: 9-track tapes (6250 bpi), CDs, 4mm and 8mm cassette tapes.

Test Case 1 Physical Electronic Media Ingest

This test verifies that all physical electronic media identified for IR-1 can be used to ingest data.

Input: Physical electronic media {9-track tape (6250 bpi), CD, 4mm cassette tape and 8mm cassette tape} containing valid data.

Output: Receipt logs, directory listings and data comparisons

Expected Results: The data is successfully ingested from the physical electronic media. The receipt logs contain entries for all of the data ingested. The directory listings contain files for all of the data ingested. The data comparisons prior to and after ingest contain no differences.

Test Case 2 Bad physical electronic media

This test verifies that physical electronic media identified for IR-1 that is corrupted can be identified and that a request for a new copy of the corrupted media is made.

Input: Corrupted physical electronic media {9-track tape (6250 bpi), CD, 4mm cassette tape and 8mm cassette tape}.

Output: Ingest error log. Request for a new copy of the corrupted media.

Expected Results: The error contains an entry for the corrupted physical media. A request is made for a new copy of the corrupted media.

4.1.2.2.8 Sequence 8 - Ingest Management

The following series of tests verify that the management of the ingest process includes gathering performance statistic, securities statistics and maintaining the proper file directory management.

Test Case 1 Performance

This test verifies that performance statistics can be gathered for the ingest process.

Input: DAN with corresponding valid data transferred electronically to an ECS DAAC.

Output: Performance statistics.

Expected Results: Performance statistics are gathered for the ingest process.

Test Case 2 Security

This test verifies that security statistics can be gathered for the ingest process for host-level access only.

Input: DAN with corresponding valid data transferred electronically to an ECS DAAC.

Output: Security statistics.

Expected Results: Security statistics are gathered for the ingest process for host-level access only.

Test Case 3 File Directory

This test verifies that file directory management is maintained for ingest staging storage only.

Input: DAN with corresponding valid data transferred electronically to an ECS DAAC. Existing directory structure.

Output: Directory listings.

Expected Results: The directory listings contain the correct file entries for the ingested data.

Test Case 4 Algorithm Storage (Staging Only)

This test verifies that the ECS can store (staging only) algorithms.

Input: Algorithm on staging disk.

Output: Directory listing of staging disk.

Expected Results: The directory listing contains the algorithm file.

4.1.3 Algorithm Integration and Test Build

The following listing identifies the test sequences within this build, its paragraph reference number and page number of the accompanying text.

4.1.3.1 Build Objectives

This build provides the capabilities to support the algorithm integration and test to be performed for IR-1. Specifically, IR-1 must be able to:

- Receive the Science Software Delivery Package via electronic transfer (ftp);
- Configuration manage the Science Software Delivery Package;
- Check the algorithm for compliance with ECS standards;
- Compile the source code and run the algorithm test using the SCF version of the toolkit.
- Compile the source code and run the algorithm test using the DAAC version of the PGS toolkit and the Planning and Data Processing System (PDPS) to verify integration into the DAAC;
- Compare the locally produced results with those supplied by the SCF;
- Send the test results to the SCF for analysis;
- Manually generate standard TRMM data products;
- Perform Science Software Delivery fault detection.

For IR-1 only a limited set of PGS capabilities are provided to support integration and test of TRMM version 1 algorithms and AM-1 Beta version algorithms. This includes a limited version of the DAAC PDPS.

4.1.3.2 Build Test Description

The algorithm integration and test capabilities required for IR-1 are tested by conducting several algorithm and test cases with representative algorithms and data. Several representative algorithm, or variations of a single algorithm, are required to fully exercise the Algorithm I&T Tools and operational procedures. For example the build test will use fully compliant algorithms as well as non compliant algorithms to verify the compliance checkers can detect non compliance with ECS standards. Similarly, erroneous data files are produced to test the data comparison tools.

The test begins with the transfer of several Science Software Delivery Packages via ftp to a representative DAAC host from the EDF. The algorithm integration and test procedures (provided with the Science Software Delivery Package) are followed to configuration manage, integrate, and test the algorithms in each package. The PDPS is used to plan the algorithm integration and test resources. Execution of the plan is a manual process. Several production plans will be setup in the database. A processing request will be entered manually to activate an appropriate production plan to generate a standard TRMM data product. A production plan consists of science algorithm inputs, control parameters, ancillary data resource validation, resource utilization, predicted processing time and simulated TRMM data. These requests can be manually activated, suspended, resumed, canceled and modified. The on-line status of requests or jobs in the processing queue will be displayed. Algorithms with known discrepancies (e.g., non compliance to ECS standards) are identified during the process. Fault detection is also tested in this build.

Dependencies (if Applicable):

- Algorithm I&T Preparatory Thread

Test Support Requirements

- Hardware:
  - HP735, SUN Sparc10, DEC 3000/300, IBM RS6000, SGI R3000 and Cray
- Software:
  - Algorithm I&T Tools (e.g., Code Checkers)
  - SCF and DAAC version of PGS Toolkit
  - Configuration Management Tool
  - Communication Software (ftp)
  - Data Comparison Tools
  - CERES and LIS data generation tool or simulated data
- Data:

The representative test data will include the following categories of files:

- Representative science algorithms
- Algorithm Output Files
- TRMM data
- Ancillary data
- Calibration coefficient files
- Processing control parameter/resource files

4.1.3.2.1 Sequence 1 - Science Software Delivery Package Receipt

The following series of tests verify that a Science Software Delivery Package can be received via ftp from a SCF to a representative DAAC host in the EDF. The Science Software Delivery Package received should include at a minimum: algorithm identification, algorithm source code, list of required inputs, calibration coefficients, processing dependencies, algorithm documentation, test procedures and test data.

Test Case 1 Successful Science Software Delivery Package

This test verifies that an Science Software Delivery Package can be transferred via ftp from a SCF to a representative DAAC host in the EDF. The Science Software Delivery Package must contain all the required information in order for the algorithm to be placed under configuration control, compiled, and executed. This test verifies that each required part of the package is transferred successfully.

Input: SCF notification of intent to deliver a Science Software Delivery Package.

Output: The entire Science Software Delivery Package to include algorithm identification, algorithm source code, algorithm documentation, list of required inputs, calibration coefficients, processing dependencies, test procedures and test data; appropriate status messages, message log file. This check for completeness is manual and a Delivery Package Evaluation report should be filed to report the completeness of the transfer. The SCF will notify the EDF that the package delivery was completed.

Expected Results: The entire Science Software Delivery Package should be transferred to the requested site, appropriate status messages should be displayed, and the message log file should be updated to reflect the requested Science Software Delivery Package was transferred to the requested location.

Test Case 2 Invalid Science Software Delivery Package Requests

This test verifies that the system responds to an invalid Science Software Delivery Package request. Errors will be inserted into the transfer request to include a request for a non-existing algorithm, a request for a transfer package from a non-existing location, and a request for a transfer package to be sent to a non-existing location. The system should respond appropriately to each invalid request.

Input: Erroneous transfer requests to include: a request for a non-existing algorithm, a request from a non-existing SCF, a request to a non-existing transfer location.

Output: Status messages, history log updates.

Expected Results: The system should respond to the invalid requests with appropriate status messages and entries in the history log.

Test Case 3 Partial Science Software Delivery Package

This test verifies that the system responds to a partial Science Software Delivery Package. Each algorithm package must contain all the required information. The check for completeness is manual and a Delivery Package Evaluation report should document any discrepancies discovered after the transfer.

Input:	SCF notification of intent to transfer a Science Software Delivery Package, network interrupt during the transfer.
Output:	Status message, history log update.
Expected Results:	The Delivery Package Evaluation report should indicate that the file transfer was incomplete and the history log should indicate an error occurred during the network transfer.

4.1.3.2.2 Sequence 2 - Configuration Management of Received Science Software Delivery Package

The following test verifies that the system can perform configuration management of the received Science Software Delivery Package. Whenever a Science Software Delivery Package is received from a SCF, the first step in processing is to place the Science Software Delivery Package under configuration management. After the package is placed under configuration management, the algorithm can then be integrated into the system. The configuration management tool is tested extensively in the Configuration Management Thread test. These tests are to ensure that the tool can be accessed from the system level and do not intend to excessively test the tool itself.

Test Case 1 Algorithm Configuration Management

This test verifies that the received Science Software Delivery Package can be placed under configuration management. The configuration management tool must be able to place each element of the transfer package into its system.

Input:	Configuration Management tool, the received algorithm package, request to place the received Science Software Delivery Package under configuration management.
Output:	Status message, history log updates.
Expected Results:	The algorithm should be placed under configuration management, appropriate status message should appear upon completion, the history log should reflect that the package is now under configuration management.

4.1.3.2.3 Sequence 3 - Algorithm Compliance Check

This test verifies that the algorithm is in compliance with ECS standards. Certain restrictions apply to the contents of each algorithm. This test will utilize a compliance checker tool to ensure the algorithm contents meet the required guidelines. A manual inventory test is also included in this sequence to validate that all required operational algorithm characteristics are present prior to scheduling algorithm test time.

<u>Test Case 1</u> <u>Algorithm in Compliance with ECS Standards.</u>

This test verifies that the received algorithm is in compliance with ECS standards. The test will utilize a POSIX checker tool to ensure the source code is within ECS guidelines.

Input: Portable Operating System Interface For Computer Environment (POSIX) Checker, algorithm with compliant source code.

Output: Status message, history log

Expected Results: The POSIX checker should report that the algorithm is within ECS standards.

<u>Test Case 2</u> <u>Algorithm Not in Compliance with ECS Standards</u>

This test verifies that the received algorithm is not in compliance with ECS standards. The test will utilize a POSIX checker to find the non compliant source code within the algorithm.

Input: Portable Operating System Interface For Computer Environment (POSIX) Checker, algorithm with non compliant source code.

Output: Status message, history log

Expected Results: The POSIX checker should report that the algorithm is not within ECS standards.

<u>Test Case 3</u> <u>Algorithm Readiness Inventory</u>

This test validates that the received algorithm characteristics are present prior to scheduling algorithm test time. This process is a manual process. The tester will record on the Inventory Log if the required characteristics are present. Any discrepancies will be noted in the Delivery Package Evaluation Report.

Input: Science Algorithm Delivery Package, Inventory Log.

Output: Delivery Package Evaluation Report.

Expected Results: The Inventory Log should include entries for the following characteristics of a received algorithm package: language used, operational impacts (e.g. algorithm software size, required resources), algorithm documentation, data handling standards, units used, models used, operational compatibility, and required metadata outputs. Any additional characteristics can be added to the log.

4.1.3.2.4 Sequence 4 - SCF Algorithm Compile

The following series of tests demonstrate that the system can compile a received algorithm.

Test Case 1 Successful Compile using SCF Version of the PGS Toolkit

This test verifies that a received algorithm can be compiled using the SCF version of the PGS toolkit. This is the first test in integrating an algorithm into the ECS system.

Input: SCF version of the PGS toolkit, science algorithm.

Output: Status messages, history log.

Expected Results: The algorithm should be successfully compiled without errors. Appropriate status messages should be generated to reflect this activity and the history log updated.

Test Case 2 Unsuccessful Compile using SCF Version

This test verifies that the system responds to error conditions detected during algorithm compilation.

Input: SCF version of the PGS toolkit, science algorithm containing compile errors.

Output: Status message, history log.

Expected Results: The system should report errors encountered during algorithm compilation and the history log should be updated to reflect these errors.

4.1.3.2.5 Sequence 5 - DAAC Algorithm Port and Compiling

The following series of tests demonstrate that the system can port the algorithm to the DAAC platform and compile it.

Test Case 1 Successful Compile using DAAC Version of the PGS Toolkit

This test verifies that a received algorithm can be ported to the DAAC platform and compiled using the DAAC version of the PGS toolkit. ANSI certified compilers/linkers are used for this test.

Input: DAAC version of the PGS toolkit, algorithm to be ported to the DAAC platform.

Output: Status message, history log.

Expected Results: The system should report on the successful porting and compilation of the science algorithm and the history log should be updated.

Test Case 2 Unsuccessful Compile using DAAC Version of the PGS Toolkit

This test verifies errors encountered during compilation are reported. ANSI certified compilers/linkers are used for this test.

Input: DAAC version of the PGS toolkit, algorithm containing compile errors.

Output: Status message, history log.

Expected Results: The system should report all compile errors encountered during compilation, and the history log should be updated to reflect such errors.

4.1.3.2.6 Sequence 6 - DAAC Algorithm Scheduling

The following series of tests demonstrate that the system can execute an algorithm using the PDPS. The sequence will also demonstrate the system can suspend, resume, and cancel the execution of a processing request. The PGS processing log will also be verified to ensure that it accounts for all processing activities. The PGS resource utilization report will be generated and verified.

Test Case 1 Successful Scheduling of Algorithm

This test verifies the acceptance of the processing requests entered manually by the tester for generating Level 1A TRMM data products.

Input: Production plans containing science algorithm inputs, control parameters, resource validation, resource utilization, predicted processing time and simulated TRMM data product will be setup in the database. A processing request will be entered manually to activate the production plan.

Output: Status message, history log, results of algorithm execution.

Expected Results: The PDPS should be able to schedule the science algorithm to execute at the DAAC and report on the schedule time, the success of execution, and the results of execution. The history log should be updated to reflect successful scheduling and execution

Test Case 2 Unsuccessful Scheduling

This test verifies that the system responds to attempts to schedule algorithms which cannot be scheduled due to data dependencies.

Input: PDPS, algorithm which cannot be scheduled due to data dependencies.

Output: Status message, history log.

Expected Results: The system should provide status messages reporting that the algorithm cannot be scheduled due to data dependencies. The history log should note that the attempt was unsuccessful.

Test Case 3 Suspending Execution of a Scheduled Processing Request

This test verifies that a scheduled processing request can be suspended. The tester will schedule a processing request to be executed. While it is executing, the tester will use the scheduling software to elect to suspend the executing processing request. A status message should be generated to indicate the processing request is suspended. Once it is suspended, the tester will verify its status.

Input: PDPS, display to observe processing request status, tool to suspend the executing processing request.

Output: Status message, display update.

Expected Results: The display should update the status of the executing processing request to suspended, status message indicating the processing request is suspended.

Test Case 4 Resume Execution of a Scheduled Processing Request

This test will continue from test case 3 above. After the processing request is suspended and its status verified on a display, the tester will then use the scheduling software to resume the processing request. A status message should be generated to indicate the processing request is resumed. Once it is resumed, the tester will verify its status.

Input: Display to observe processing request status, tool to resume the suspended processing request.

Output: Status message, display update.

Expected Results: The display should update the status of the suspended processing request to executing, status message indicating the processing request is executing.

Test Case 5 Canceling Execution of a Scheduled Processing Request

This test verifies that a scheduled processing request can be canceled. The tester will schedule a processing request to be executed. While it is executing, the tester will use the scheduling software to elect to cancel the executing processing request. A status message should be generated to indicate the processing request is canceled. Once it is canceled, the tester will verify its status.

Input: Display to observe processing request status, tool to cancel the executing processing request.

Output: Status message, display update.

Expected Results: The display should update the status of the executing processing request to canceled, status message indicating the processing request is canceled.

Test Case 6 Queue

This test verifies the capability of changing the queue position of processing requests. The tester will manually change the position of a processing request in a queue after it has been scheduled for processing.

Input: Manually enter processing requests to activate an appropriate production plan to generate a data product. Change position of the requests in the queue for process.

Output: The tester can view the on-line status of requests in the processing queue. There will also be status in the message log.

Expected Results: Changing the queue for processing requests will be logged in the processing log.

Test Case 7 Priority

This test verifies the capability to change request priorities. The tester will change the priority of a scheduled processing request. The system should verify this input and log the changed priority.

Input: Manually enter requests to activate an appropriate production plan to generate data product. Change the priorities of the requests in the processing queue.

Output: The tester can view the on-line status of requests in the processing queue. There will also be status in the message log.

Expected Results: Changing priorities of requests in the processing queue will be logged in the processing log.

Test Case 8 Invalid Cancellation

This test verifies users authentication to cancel processing requests other than those generated by themselves. Several processing request will be scheduled by various testers. Then, the tester will try to cancel a request other than his own. The system should not allow this action and log the attempted transaction.

Input: Manually enter requests to activate appropriate production plans. Attempt to cancel the request of another tester after it begins execution.

Output: The tester can view the on-line status of requests in the processing queue. There will also be a status of the request in the message log.

Expected Results: The request will not be terminated and an appropriate message will be logged.

Test Case 9 Invalid Priority Change

This test verifies thar error messages are recorded in the processing log as a result of invalid change in request priority. Certain priorities will be reserved for identified requests. In this test, the tester will attempt to change the priority of his scheduled request beyond its allowed range. The system should respond to this request and log the attempt.

Input: Manually enter a request to activate an appropriate production plan. Modify the priority of the request after it begins execution.

Output: The tester can view the on-line status of requests in the processing queue. There will also be a status of the request in the message log.

Expected Results: The priority will not be changed and an appropriate message will be logged.

Test Case 10 Invalid Change of Queue

This test verifies that error messages are recordedin the processing log as a result of invalid change of queue position. The tester will attempt to move his scheduled request in the queue to a position outside of the allowed range. The system should respond to this attempt.

Input: Manually enter a few requests to activate an appropriate production plan. Change the queue position of the requests that is currently being executed.

Output: The tester can view the on-line status of requests in the processing queue. There will also be a status for each request in the message log.

Expected Results: The queue position will not be changed and an appropriate message will be logged.

Test Case 11 PGS Processing Log Verification

This test will verify that the PGS processing log accounts for all the scheduling activities in this test sequence. The tester will print the processing log after completing test cases 1 through 10. The processing log should contain correct information for the activities which occurred in this test sequence.

Input: Test cases 1-10 of this sequence.

Output: PGS processing log print out.

Expected Results: The PGS processing log should reflect all activities which occurred during this test sequence.

Test Case 12 PGS Resource Utilization Report Verification

After test cases 1-10 have been successfully executed, the PGS resource utilization report will be printed and verified. The report should show the resources used to execute the processing requests.

Input:	Test cases 1-10 of this sequence.
Output:	PGS resource utilization report print out.
Expected Results:	The PGS resource utilization report should reflect all activities which occurred during this test sequence.

4.1.3.2.7 Sequence 7 - Planning and Data Processing Faults

The following series of tests verify that the system management framework properly detects a hardware fault within the Planning and Data Processing sub-systems and the event logging utility properly records all algorithm sequences planned, queued, and processed. The Planning and Data Processing hardware will consist of three (most likely 2) computing machines, one for the planning application, one for the process queuing, and the other for data processing. Each of the machines will be monitored by the system management framework utility and all system activities will be recorded within the event logs. To complete these tests, many algorithm jobs must be processing, and the planning and queuing of additional algorithm tests must be in progress.

<u>Test Case 1</u> <u>Data Processing Fault</u>

This test verifies that the system management framework tool detects and locates the data processing machine fault and properly logs all events.

Input:	Algorithm jobs. System management framework utility. Turn power off to the data processing machine.
Output:	System management framework alerts tester of hardware fault. Algorithm processing halted. Planning and queuing still in progress.
Expected Results:	Event log records all activities of planning/queuing machine and data processing machine. Event log should also list algorithms that were processing at the moment the data processing machine went down, and the completed algorithm processes.

<u>Test Case 2</u> <u>Queuing Fault</u>

This test verifies that the system management framework tool detects and locates the queuing machine fault and properly logs all events.

Input:	Algorithm jobs. System management framework utility. Turn power off on the queuing machine.
Output:	System management framework alerts tester of hardware fault. Algorithm processing and planning continues. Queuing of algorithms halts.

Expected Results: Event log records all activities of planning/queuing and data processing machines. Event log should also list algorithms queued for processing, and which jobs were sent for processing.

Test Case 3 Planning Fault

This test verifies that the system management framework tool detects and locates the planning machine fault and properly logs all events.

Input: Algorithm jobs. System management framework utility. Turn power off on the computer running the planning application.

Output: System management framework alerts tester of hardware fault. Algorithm queuing and processing continues while planning halts.

Expected Results: Event log records all activities of planning/queuing and data processing machines. Event log should also list algorithm jobs planned and complete.

Test Case 4 Verification of Event Log

This test verifies that the event log file records all system activities and algorithm jobs. Manually, the tester will review the event log file to verify that tests 1-3 were completely recorded. The event log should contain correct information for the activities which occurred in this test sequence.

Input: Test cases 1-3 of this sequence.

Output: Event log file print out.

Expected Results: The event log file should reflect all activities which occurred during this test sequence.

4.1.3.2.8 Sequence 8 - Results Comparison

The following series of tests compare the local results of running a received algorithm with the expected results provided in the Science Software Delivery Package. A data comparison tool will be used to confirm the algorithm results.

Test Case 1 Successful Comparison

This test verifies that the expected results provided in the Science Software Delivery Package match those generated during the actual execution of the algorithm.

Input: Comparison tool, expected results, actual results.

Output: Status message, history log.

Expected Results: The system should report on the successful comparison of the expected results with the actual results.

Test Case 2 Unsuccessful Comparison

This test verifies that the system reports differences when the expected results provided in the Science Software Delivery Package do not match those generated during the actual execution of the algorithm.

Input: Comparison tool, expected results, actual results (modified to provide errors).

Output: Status message, history log.

Expected Results: The system should report on the unsuccessful comparison of the expected results with the actual results.

4.1.3.2.9 Sequence 9 - Send Test Results to SCF

The following series of tests send the algorithm test results to the SCF for analysis. The system should also be able to handle errors which may occur during this process.

Test Case 1 Successful Transfer

This test verifies that the system is able to send the algorithm test results to the SCF for analysis. The results should include: algorithm identification, test times, processor identification and test results.

Input: Actual test results.

Output: Status message, history log.

Expected Results: The system should report on the successful transfer of the test results to the SCF.

Test Case 2 Unsuccessful Transfer

This test verifies that the system is able to respond to errors while attempting to send the algorithm test results to the SCF for analysis. An error will be induced to make the system fail during the transfer.

Input: Actual test results.

Output: Status message, history log.

Expected Results: The system should report on the failed attempt to transfer the test results to the SCF.

Test Case 3 Partial Transfer

This test verifies that the system is able to detect errors in content while trying to send the algorithm test results to the SCF for analysis. The results should include: algorithm identification, test times, processor identification and test results. The system should report if any of these elements are omitted from the transfer.

Input: Actual test results (modified to not contain all required elements).

Output: Status message, history log.

Expected Results: The system should report on the failed attempt due to incomplete packaging of the test results.

4.1.3.2.10 Sequence 10 - Standard TRMM Data Products

The following series of tests verify the capabilities of the Initial PDPS to acquire ancillary data to generate standard TRMM data products at Level 1B, Level 2, Level 3. and Level 4. These tests are dependent on successful representative science algorithms with control parameters, ancillary data and simulated TRMM instrument data for requesting a standard data product. Data availability and erroneous data are also verified.

Test Case 1 Level 1B TRMM Data Product

This test verifies the capability to accept the processing request for generating Level 1B data product.

Input: Manually enter requests to activate an appropriate production plan for generating Level 1B TRMM data product.

Output: The tester can view the on-line status of requests in the processing queue. There will also be status of requests in the message log.

Expected Results: An acceptance of request will be logged.

Test Case 2 Level 2 TRMM Data Product with Terrain Map Database

This test verifies the capability to acquire a digital terrain map database as an input for a request for generating Level 2 TRMM data products.

Input: Production plans containing science algorithm inputs, control parameters, resource validation, resource utilization, predicted processing time, digital terrain map database and simulated TRMM data product will be setup in the database. A processing request will be entered manually to activate this production plan for generating Level 2 TRMM data product.

Output: The tester can view the on-line status of requests in the processing queue. There will also be status of requests in the message log.

Expected Results: An acceptance of request will be logged.

Test Case 3 Level 2 TRMM Data Product with Land/Sea Database

This test verifies the capability to acquire a land/sea databases as an input for a request for generating Level 2 TRMM data products.

Input: Production plans containing science algorithm inputs, control parameters, resource validation, resource utilization, predicted processing time, land/sea databases and simulated TRMM data product will be setup in the database. A processing request will be entered manually to activate this production plan for generating Level 2 TRMM data product.

Output: The tester can view the on-line status of requests in the processing queue. There will also be a status of request in the message log.

Expected Results: An acceptance of request will be logged.

Test Case 4 Level 3 TRMM Data Product

This test verifies the capability to acquire climatology databases as an input for a request for generating Level 3 TRMM data products.

Input: Production plans containing science algorithm inputs, control parameters, resource validation, resource utilization, predicted processing time, climatology databases and simulated TRMM data product will be setup in the database. A processing request will be entered manually to activate this production plan for generating Level 3 TRMM data product.

Output: The tester can view the on-line status of requests in the processing queue. There will also be status of requests in the message log.

Expected Results: An acceptance of request will be logged.

Test Case 5 Level 4 TRMM Data Product

This test verifies the capability to acquire digital political map databases as an input for a request for generating Level 4 TRMM data products.

Input: Production plans containing science algorithm inputs, control parameters, resource validation, resource utilization, predicted processing time, digital political map databases and simulated TRMM data product will be setup in the database. A processing request will be entered manually to activate this production plan for generating Level 4 TRMM data product.

Output: The tester can view the on-line status of requests in the processing queue. There will also be status of requests in the message log.

Expected Results: An acceptance of request will be logged.

Test Case 6 Missing Data

This test verifies that the processing request will not be initiated until data is available. Before the products generation time is reached, the tester will remove the necessary Level 0 data. When the execution begins, the production should be suspended due to data availability. The tester will then replace the data which was removed. At this time, the tester will resume execution of the product generation.

Input: A few production plans containing science algorithm inputs, control parameters, resource validation, resource utilization, predicted processing time and simulated Level 0 data will be setup in the database. Method to remove and re-install data.

Output: The tester can view the on-line status of the request in the processing queue. There will also be a status of request in the log.

Expected Results: The request will not be processed until the data become available. These activities will be logged.

Test Case 7 Erroneous Data

This test verifies that the processing request will not be initiated if the arrival of data is incorrect to generate Level 1A TRMM data product. Erroneous data will need to be inputted into the system. This data should be selected by the production generation program. When the data is processed, the error should be detected and product generation canceled.

Input: Production plans containing science algorithm inputs, control parameters, resource validation, resource utilization, predicted processing time and simulated Level 2 data will be setup in the database. Enter a processing request to activate an appropriate production plan for generating Level 1A data product. Incorporate erroneous data into the data base.

Output: The tester can view the on-line status of the request in the processing queue. There will also be a status of request in the log.

Expected Results: The request will not be processed once the erroneous data is discovered. This activity will be logged.

4.1.3.2.11 Sequence 11 - Science Software Delivery Faults

The following series of tests verify that the system management framework tool will detect file transfer faults between science computing facilities and DAACs. The system management framework tool should detect and locate faults caused from LAN/WAN faults, host machine faults, and ftp process termination.

Test Case 1 Local Area Network Fault

This test verifies that the system management framework tool will properly detect and locate the network fault that occurs during the transfer of Science Software Delivery Packages. The network fault will occur within the LAN of a representative DAAC host in the EDF. This test verifies that proper alarms are displayed and the fault is recorded in event logs.

Input: SCF notification of intent to deliver a Science Software Delivery Package. Root map of system management framework tool displayed on tester's machine. Turn power off on gateway/router into DAAC LAN.

Output: File transfer is terminated. DAAC symbol on root map of System Management Framework has turned RED.

Expected Results: Software delivery terminated. Traversing through the system management framework maps the tester should be directed to the faulty gateway/router. Log file should record all system and tester activities.

Test Case 2 Wide Area Network Fault

This test verifies that the system management framework tool will properly detect and locate the network fault that occurs during the transfer of Science Software Delivery Packages. The tester will simulate the network fault within the connection of the WAN to the LAN.

Input: SCF notification of intent to deliver a Science Software Delivery Package. Root map of system management framework tool displayed on tester's machine. Connection of WAN to LAN disconnected.

Output: File transfer terminated. Internet symbol on root map of system management framework has turned RED.

Expected Results: Software delivery terminated. Traversing through the system management framework maps the tester should be directed to the problem. Log file should record all system and tester activities.

Test Case 3 Host Machine Fault

This test verifies that the system management framework tool will properly detect and locate the host machine within the EDF that caused the simulated network fault. The host machine within the EDF where the software is being delivered will be shutdown, during this process the transfer should be terminated and the system management framework should notify tester of fault.

Input: SCF notification of intent to deliver a Science Software Delivery Package. Root map of system management framework tool displayed on testers machine. Shut down host machine where software delivery is targeted.

Output: File transfer terminated. DAAC symbol of root map of system management framework has turned YELLOW.

Expected Results: Software delivery terminated. Traversing through the system management framework maps the tester should be directed to the fault host machine. Log file should record all system and tester activities.

<u>Test Case 4</u> <u>File Transfer Termination</u>

This test verifies that the system management framework tool will properly detect a file transfer termination. Testing includes monitoring the system management framework tool while the file transfer process is located and terminated.

Input: SCF notification of intent to deliver a Science Software Delivery Package. Root map of system management framework tool displayed on tester's machine. Locate and terminate file transfer process from SCF.

Output: File transfer terminated abruptly. Internet symbol of root map of system management framework has turned YELLOW.

Expected Results: Software delivery terminated. Traversing through the system management framework maps the tester should be directed to the transfer fault. Log file should record all system and tester activities

<u>Test Case 5</u> <u>Verification of Log File</u>

This test verifies that the log file records all system and tester activities. Manually, the tester will review the log file to verify that tests 1-4 were completely recorded. The log should contain correct information for the activities which occurred in this test sequence.

Input: Test cases 1-4 of this sequence.

Output: Log file print out.

Expected Results: The log file should reflect all activities which occurred during this test sequence.

4.1.3.2.12 Sequence 12 - Algorithm/Method Toolkit Regression Tests

The following series of tests serve as a regression test for the toolkit functionality in the Algorithm Integration & Test Preparatory build which feeds into the Algorithm Integration & Test Build. These tests assess algorithms that utilize PGS Toolkits. Algorithms developed and accepted as valid by the SCF, using the SCF Toolkit in the SCF environment, are run in the DAAC environment. The algorithm/method is tested using the data sets (or references) delivered with the software. Test results are analyzed to validate the correctness of the data products.

Test Case 1 Geolocation/Geocoordination Conversion Test

This test verifies that an algorithm developed and deemed valid at the SCF, will run correctly giving the same algorithm output as produced in the SCF when run at the DAAC. Representative SCF algorithms that utilizes PGS Toolkit functions for geolocation/geocoordinate transformations are run on all approved DAAC platforms. The results are analyzed to determine the scientific correctness of the data products.

Input: Algorithms developed at the SCF and SCF /DAAC Toolkits. Access to all subroutines and libraries containing data needed to perform data conversions. These algorithms will use the PGS Toolkit for geolocation/geocoordinate transformations. The algorithm is first run at the DAAC using the PGS SCF Toolkit version. Then the algorithm is run at the DAAC using the DAAC Toolkit version.

Output: Algorithm status, algorithm product output for both the SCF Toolkit and the DAAC Toolkit.

Expected Results: The product outputs from the SCF and DAAC Toolkits are analyzed for correct scientific product content. There should be similar output from the SCF and DAAC Toolkits. Outputs from the SCF and DAAC Toolkits may be examined using a proven data comparison tool.

Test Case 2 Time/Date Conversion Test

This test verifies that an algorithm developed and deemed valid at the SCF, will run correctly giving the same algorithm output as produced in the SCF when run at the DAAC. Representative SCF algorithms that utilizes PGS Toolkit functions for time/date conversions are run on all approved DAAC platforms. The results are analyzed to determine the scientific correctness of the data products.

Input: Algorithms developed at the SCF and SCF /DAAC Toolkits. Access to all subroutines and libraries containing data needed to perform data conversions. These algorithms will use the PGS Toolkit for time/date conversions. The algorithm is first run at the DAAC using the PGS SCF Toolkit version. Then the algorithm is run at the DAAC using the DAAC Toolkit version.

Output: Algorithm status, algorithm product output for both the SCF Toolkit and the DAAC Toolkit.

Expected Results: The product outputs from the SCF and DAAC Toolkits are analyzed for correct scientific product content. There should be similar output from the SCF and DAAC Toolkits. Outputs from the SCF and DAAC Toolkits may be examined using a proven data comparison tool.

Test Case 3 Calibration Coefficients and Algorithm Update Test

This test verifies the DAAC's ability to receive and update science software using new or modified (updated) algorithms/calibration coefficients. This test will evaluate the ability of the DAAC Toolkit environment to modified existing science software when an algorithm/coefficient update is deemed necessary by the SCF. An algorithm proven to be a valid algorithm from a previous test (Geolocation/Geocoordination Conversion Test or Time/Date Conversion Test) is modified and run at the SCF. The results are recorded. The same algorithm is modified at the DAAC using updated procedures and data received from the SCF. The results of the SCF output are compared to the DAAC output.

Input: Algorithms validated in previous tests such as the Geolocation/Geocoordination Conversion Test or Time/Date Conversion Test. New or updated algorithms and calibration coefficients. Access to all subroutines and libraries containing data needed to perform data conversions. Access to a script editor to make changes to the algorithms.

Output: Algorithm status, algorithm product output for both the SCF Toolkit and the DAAC Toolkit.

Expected Results: The product outputs from the SCF and DAAC Toolkits are analyzed for correct scientific product content. There should be similar output from the SCF and DAAC Toolkits. Outputs from the SCF and DAAC Toolkits may be examined using a proven data comparison tool.

4.1.3.2.13 Sequence 13 - ECS User Community

The following tests verify the ECS’s capabilities to exchange necessary data with the Science User Community to replace and/or update the algorithms running at the appropriate DAAC.

Test Case 1 Software Problem Reports

This test verifies the ECS’s capability to send a Software Problem Report to the ECS Science Community. The Software Problem Report will contain the reason for updating and/or replacing the algorithms running at an appropriate DAAC.

Input: Write a Software Problem Report at an appropriate DAAC. This report should contain the reason for updating the algorithm running at that DAAC. Transmit this report to the ECS Science Community.

Output: The ECS Science Community will receive the Software Problem Report from the DAAC.

Expected Results: The content of the Software Problem Report received at the science community should be exactly the same as it was sent from the DAAC.

Test Case 2 Integration Support Request

This test verifies the ECS's capability to send for receiving an integration support request from the ECS Science Community. An integration support request will be to replace and/or update the algorithms running at an appropriate DAAC.

Input: Management policies and procedures integration support guidelines from the EDF.

Output: Management policies and procedures manual from GSFC, MSFC, and LaRC.

Expected Results: Guidelines within document are current and identical to that copy
held at the EDF.

4.1.3.2.14 Sequence 14 - Operational Availability and PGS Capacity

The following test case will verify that each computer providing product generation will have an operational availability, I/O to Intermediate Storage and PGS capacity required to support this release.

Test Case 1 Operational Availability

An analysis will be performed for the appropriate hardware and software to ensure that each computer providing product generation shall have an operational availability of 0.95 at a minimum.

Input: Support documentation for analysis

Output: A detailed analysis report

Expected Results: Presented in RMA Analysis Report

Test Case 2 I/O to Intermediate Storage

An analysis will be performed for appropriate hardware and software to ensure that the PGS has the capacity to support I/O to intermediate storage as required by individual science algorithms.

Input: Support documentation for analysis

Output: A detailed analysis report

Expected Results: Presented in RMA Analysis Report

Test Case 3 Multiple Passes over Input Product

An analysis will be performed for appropriate hardware and software to ensure that the PGS has the capacity to support multiple passes over input products as required by individual science algorithms.

Input: Support documentation for analysis

Output:	A detailed analysis report
Expected Results:	Presented in RMA Analysis Report

4.1.4 Interim Release 1 Build

The following listing identifies the test sequences within this build, its paragraph reference number and page number of the accompanying text.

4.1.4.1 Build Objectives

The objective of the Interim Release 1 build is to provide global monitoring of the active hardware and software of the system. The System Management Framework Tool (HP OpenView) provides the capability of viewing the system activity through maps, signals and messages. This build is also expected to verify the readiness of the external interfaces between SCF, ADCs and the DAACs. The IR-1 build also provides network statistics for evaluation and the analysis of network performance. Functional capabilities tested in previous builds within this release will not be duplicated here. The functions being integrated into the earlier tested capabilities will be demonstrated here in order to fully round out all of this release’s capabilities.

4.1.4.2 Build Test Description

The System Management tool will be active and a list of active hardware components will be used. On the screen, all active hardware and application will be displayed by the System Management Tool (HP OpenView). Ability to properly detect and monitor all the hardware will be tested for GSFC, MSFC, LaRC and the EDF. Activation of HP OpenView Network Node Manager will properly detect and locate hardware faults within the LAN/WAN network. The following situations will be used for testing for hardware failure:

- power loss
- termination of hardware monitoring
- loss of network connection

Testing will be performed on the System Management Framework Tool's ability to detect network fault caused:

- gateway/router power loss
- gateway /router monitoring process terminated
- network disconnected from gateway/router

The ability of the System Management Framework Tool to properly detect and locate network faults caused by computer outages will also be tested. The following situations will be used for the loss of a connection:

- loss of power
- terminals disconnected from network
- termination of monitoring process
- termination of operating system processing
- hard disk failure
- Exceeds the memory threshold limit
- CPU threshold fault.

Proper detection of abnormally terminated software application and software application monitoring processes will also be tested. Software process monitoring will be tested for the GSFC, MSFC, LaRC and EDF sites software. System Management framework tool will detect when a multi-processing terminated abruptly.

Selected real or simulated algorithms will be used for algorithm integration activities. External interface readiness will be tested for a two way or one way transfer of algorithms between the following :

- TRMM-1 algorithm to/from SCF to/from GSFC DAAC
- TRMM-1 algorithm to/from SCF to/from MSFC DAAC
- TRMM-1 algorithm to/from SCF to/from LaRC DAAC
- AM-1 algorithm to/from SCF to/from EDC DAAC
- algorithm transfer from ADC to GSFC, MSFC, LaRC DAAC

(The ingest build previously demonstrated the interface readiness between TSDIS and SDPF & EDF and DAACs)

A series of instances may occur where network performance needs to be evaluated. The ability to display network statistics by GUI, and output logging for network statistics and local site statistics will be required to evaluate the network. Therefore these capabilities will be tested for each site.

Dependencies: (If Applicable)

- System Access Build
- Algorithm Integration & Test Build
- Ingest Build
- ECS Administrative Thread

Test Support Requirements

- Hardware:

  - Workstation (mid-size)

- Software:

  - Xwindows software
  - Automated GUI Test Tool
  - HP OpenView
  - SCF and DAAC versions of PGS Toolkit
  - Data Comparison Tools
  - ClearCase
  - POSIX Checkers
  - ANSI certified compilers/linkers

- Data:

  - Selected V0 data (HDF and native formatted)
  - TRMM instrument CERES and LIS simulated Level 0 data products
  - Representative Science Algorithms written in C and FORTRAN
  - Calibration coefficient files
  - Ancillary data or simulated ancillary data

4.1.4.2.1 Sequence 1 - External Interface Readiness

The following series of tests verify the availability of the external interfaces. The various interface locations and interaction vary as well as the expected information being passed.

Test Case 1 TRMM/AM-1 Algorithms from SCF to/from GSFC DAAC

This test verifies the interface between SCF's and GSFC for Algorithm ingest activities.

Input: Real or simulated algorithms.

Output: Receipt of transmitted data are identical to those sent.

Expected Results: Confirmation by the GSFC of receipt of algorithm.

Test Case 2 TRMM/AM-1 Algorithms from SCF to/from MSFC DAAC

This test verifies the interface between SCF's and MSFC for Algorithm ingest activities.

Input: Real or simulated algorithms.

Output: Receipt of transmitted data are identical to those sent.

Expected Results: Confirmation by the MSFC of receipt of algorithm.

Test Case 3 TRMM/AM-1 Algorithms from SCF to/from LaRC DAAC

This test verifies the interface between SCF's and LaRC for Algorithm ingest activities.

Input: Real or simulated algorithms.

Output: Receipt of transmitted data are identical to those sent.

Expected Results: Confirmation by the LaRC of receipt of algorithm.

Test Case 4 AM-1 Algorithms from SCF to/from EDC DAAC

This test verifies the interface between SCF's and EDC for Algorithm ingest activities.

Input: Real or simulated algorithms.

Output: Receipt of transmitted data are identical to those sent.

Expected Results: Confirmation by EDF of receipt of algorithm.

Test Case 5 ADC to GSFC DAAC

This test verifies the one-way interface between NOAA and GSFC for NOAA satellite and NMC ancillary data ingest activities.

Input: Real or simulated ancillary data files.

Output: Receipt of transmitted data are identical to those sent.

Expected Results: Confirmation by GSFC of receipt of ancillary data files.

Test Case 6 ADC to MSFC DAAC

This test verifies the one-way interface between NOAA and MSFC for NOAA satellite and NMC ancillary data ingest activities.

Input: Real or simulated ancillary data files.

Output: Receipt of transmitted data are identical to those sent.

Expected Results: Confirmation by the MSFC of receipt of ancillary data files.

Test Case 7 ADC to LaRC DAAC

This test verifies the one-way interface between NOAA and LaRC for NOAA satellite and NMC ancillary data ingest activities.

Input: Real or simulated ancillary data files.

Output: Receipt of transmitted data are identical to those sent.

Expected Results: Confirmation by LaRC of receipt of ancillary data files.

4.1.4.2.2 Sequence 2 - Monitoring of Hardware/Software/User Environment

The following series of tests verify that the System Management Framework tool will properly detect all hardware components active on the system at all times. These tests will confirm that all hardware, specific to the system, at GSFC, MSFC, LaRC, and the EDF are on-line and accessible through the System Management Framework tool. Testing will include contacting the DAAC liaisons or System Administrators and receiving a list of all hardware (PCs, workstations, minis, main-frames, archives, gateways/routers, printers, modems, etc.) currently active and in-use at each site.

<u>Test Case 1</u> <u>GSFC Hardware Confirmation</u>

This test verifies that the System Management Framework tool properly detects and monitors all hardware at Goddard Space Flight Center.

Input: GSFC submap of System Management Framework tool is active in the tester's display. Contact DAAC liaison and receive a list of all hardware components active and in-use.

Output: GSFC submap of System Management Framework tool displays all hardware active at GSFC. Listing of all hardware active and in-use at GSFC.

Expected Results: Hardware displayed in GSFC submap of System Management Framework tool matches/confirms hardware that is obtained from the DAAC liaison.

<u>Test Case 2</u> <u>MSFC Hardware Confirmation</u>

This test verifies that the System Management Framework tool properly detects and monitors all hardware at Marshall Space Flight Center.

Input: MSFC submap of System Management Framework tool is active in the tester's display. Contact DAAC liaison and receive a list of all hardware components active and in-use.

Output: MSFC submap of System Management Framework tool displays all hardware active at MSFC. Listing of all hardware active and in-use at MSFC.

Expected Results: Hardware displayed in MSFC submap of System Management Framework tool matches/confirms hardware that is obtained from the DAAC liaison.

<u>Test Case 3</u> <u>LaRC Hardware Confirmation</u>

This test verifies that the System Management Framework tool properly detects and monitors all hardware at Langley Research Center.

Input: LaRC submap of System Management Framework tool is active in the tester's display. Contact DAAC liaison and receive a list of all hardware components active and in-use.

Output: LaRC submap of System Management Framework tool displays all hardware active at LaRC. Listing of all hardware active and in-use at LaRC.

Expected Results: Hardware displayed in LaRC submap of System Management Framework tool matches/confirms hardware that is obtained from the DAAC liaison.

Test Case 4 EDF Hardware Confirmation

This test verifies that the System Management Framework tool properly detects and monitors all hardware at the ECS Development Facility.

Input: EDF submap of System Management Framework tool is active in the tester's display. Contact System Administrator and receive a list of all hardware components active and in-use.

Output: EDF submap of System Management Framework tool displays all hardware active at EDF. Listing of all hardware active and in-use at EDF.

Expected Results: Hardware displayed in EDF submap of System Management Framework tool matches/confirms hardware that is obtained from the System Administrator.

The following series of tests verifies that the System Management Framework Tool, HP OpenView Network Node Manager, will properly detect and locate hardware faults (peripherals, archives, printer, etc.) that are connected within the LAN/WAN network. Hardware faults may be caused by power loss, terminating the process that monitors the hardware, or a network disconnect from the hardware. All fault events should be recorded in a problem log.

Test Case 5 Hardware Power Loss

This test verifies that the System Management Framework tool will detect a hardware fault caused from a power loss. Testing includes monitoring the System Management Framework tool while the power loss to the hardware occurs. A System Management Framework window will be displayed on the workstation/PC where the tester is located. (Test will be verified at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Turn power off on hardware (peripherals, archives, etc.).

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the inactive piece of hardware (indicated by color of RED).

Test Case 6 Hardware Monitoring Process Terminated

This test verifies that the System Management Framework tool will detect and locate a hardware "fault" when the monitoring process of the specific piece of hardware has been terminated. Testing includes monitoring the System Management Framework tool while the monitoring process is terminated, therefore, a System Management Framework window will be displayed on the workstation/PC where the tester is located. (Test will be verified at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Tester will locate and terminate/"kill" the process that monitors the hardware.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the "faulty" piece of hardware (indicated by color RED).

Test 7 Network Disconnect from Hardware (Peripherals, Printers, Archives, etc.)

This test verifies that the System Management Framework tool will detect and locate a network disconnect from a piece of hardware. Testing includes monitoring the System Management Framework tool while the network disconnect occurs, therefore, a System Management Framework window will be displayed on the workstation/PC where the tester is located. (Test will be verified at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Disconnect network connect from hardware.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW/marginal status.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the "faulty" network interface (indicated by RED/critical status color) on the piece of hardware where the network was initially disconnected.

The following series of tests verify that the System Management Framework Tool, HP OpenView Network Node Manager, will properly detect and locate network faults caused by gateway/router outages. Gateway/Router outages may be caused by power loss or by terminating the process that monitors the gateway/router.

Test Case 8 — Gateway/Router Power Loss

This test verifies that the System Management Framework tool detects and locates a network fault caused from a gateway/router power loss. Testing includes monitoring the System Management Framework tool while the power loss occurs.

A System Management Framework window will be displayed on the workstation where the tester is located. (Test will be verified at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Turn power off on gateway/router.

Output: Internet symbol on Root map of System Management Framework has turned RED.

Expected Results: Traversing through the Internet submaps, following the RED/critical status symbols, the tester should be directed to the faulty gateway/router (indicated by color of RED).

Test Case 9 — Gateway/Router Monitoring Process Terminated

This test verifies that the System Management Framework tool detects and locates a possible fault with a gateway/router. Testing includes displaying the System Management Framework tool, while the tester terminates the process that monitors the gateway/router activities. (Test will be verified at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Tester will locate and "kill" the process that monitors the gateway/router.

Output: Internet symbol on Root map of System Management Framework has turned RED.

Expected Results: Traversing through the Internet submaps, following the RED/critical status symbols, the tester should be directed to the faulty gateway/router (color should be RED).

Test Case 10 — Network Disconnect from Gateway/Router

This test verifies that the System Management Framework tool detects and locates a network fault caused from a network disconnect to the gateway/router. Testing includes monitoring the System Management Framework tool while the network disconnect occurs. A System Management Framework window will be displayed on the workstation where the tester is located. (Test will be verified at each site.)

Input:	Root map of System Management Framework window will be active in tester's display. Disconnect network connection from gateway/router.
Output:	Internet symbol on Root map of System Management Framework has turned RED.
Expected Results:	Traversing through the Internet submaps, following the RED/critical status symbols, the tester should be directed to the network interface of the gateway/router. (This interface should be RED/critical.)

The following series of tests verify that the System Management Framework tool, HP OpenView Network Node Manager, will properly detect and identify the software applications that have been abnormally terminated. Software application termination may be caused by terminating the actual software application itself or by terminating the monitoring process from the System Management Framework tool. All fault events should be recorded in a problem log.

<u>Test Case 11</u> <u>Software Application Termination</u>

This test verifies that the System Management Framework tool will properly detect and locate the software application that was terminated abnormally. Testing includes monitoring the System Management Framework tool while the abnormal termination of the software application occurs, therefore, a System Management Framework window will be displayed on the workstation/PC where the tester is located. (Test will be verified at each site.)

Input:	Root map of System Management Framework window will be active in the tester's display. Tester will locate and terminate/"kill" the process that controls the software application.
Output:	Internet symbol on Root map of System Management Framework tool has turned YELLOW.
Expected Results:	Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the test should be directed to the computer where the application was running (computer will be RED in color) and the application itself.

<u>Test Case 12</u> <u>Software Application Monitoring Process Termination</u>

This test verifies that the System Management Framework tool detects and locates the terminated software application when the monitoring process of the application has been terminated. Testing includes monitoring the System Management Framework tool while the monitoring process is terminated, therefore, a management window will be displayed on the workstation/PC where the tester is located. (Test will be verified at each site.)

Input:	Root map of System Management Framework window will be active in the tester's display. Tester will locate and terminate/"kill" the process that monitors the software application.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the computer where the application was running (computer will be RED in color) and the application itself.

The following tests verifies that the Management Information Base (MIB) objects created within the System Management Framework tool to detect and monitor all software processes (GUIs, OSs, FTPs, etc.) are active on the system at all times. These tests will confirm that all software processes, specific to the system, at GSFC, MSFC, LaRC, and the EDF are monitored through the System Management Framework tool. Testing will include contacting the DAAC liaisons or System Administrators and receiving a list of all software applications and processes currently active at each site.

Test Case 13 GSFC Software Process Monitoring

This test verifies that the MIB created within the System Management Framework tool properly monitors all software processes at Goddard Space Flight Center.

Input: GSFC submap of System Management Framework tool is active in the tester's display. Contact DAAC liaison and receive a list of all software processes that are active and should be monitored.

Output: GSFC submap of System Management Framework tool displays all software processes active at GSFC. Listing of all software processes active at GSFC.

Expected Results: Software processes displayed in GSFC submap of System Management Framework tool matches/confirms software processes listed from DAAC liaison.

Test Case 14 MSFC Software Process Monitoring

This test verifies that the MIB created within the System Management Framework tool properly monitors all software processes at Marshall Space Flight Center.

Input: MSFC submap of System Management Framework tool is active in the tester's display. Contact DAAC liaison and receive a list of all software processes that are active and should be monitored.

Output: MSFC submap of System Management Framework tool displays all software processes active at MSFC. Listing of all software processes active at MSFC.

Expected Results: Software processes displayed in MSFC submap of System Management Framework tool matches/confirms software processes listed from DAAC liaison.

Test Case 15 LaRC Software Process Monitoring

This test verifies that the MIB created within the System Management Framework tool properly monitors all software processes at Langley Research Center.

Input: LaRC submap of System Management Framework tool is active in the tester's display. Contact DAAC liaison and receive a list of all software processes that are active and should be monitored.

Output: LaRC submap of System Management Framework tool displays all software processes active at LaRC. Listing of all software processes active at LaRC.

Expected Results: Software processes displayed in LaRC submap of System Management Framework tool matches/confirms software processes listed from DAAC liaison.

Test Case 16 EDF Software Process Monitoring

This test verifies that the MIB created within the System Management Framework tool properly monitors all software processes at the ECS Development Facilities.

Input: EDF submap of System Management Framework tool is active in the tester's display. Contact System Administrator and receive a list of all software processes that are active and should be monitored.

Output: EDF submap of System Management Framework tool displays all software processes active at EDF. Listing of all software processes active at EDF.

Expected Results: Software processes displayed in EDF submap of System Management Framework tool matches/confirms software processes listed from System Administrator.

The following series of tests verify that the System Management Framework Tool, HP OpenView Network Node Manager, will properly detect and locate network faults caused by computer (PCs, workstations, mini's, main frames) outages. Computer outages may be caused by power loss, terminating monitoring process, or network disconnect from the computer itself. All fault events should be recorded in a problem log.

Test Case 17 Computer Power Loss

This test verifies that the System Management Framework tool detects and locates a network fault caused from a computer power loss. Testing includes monitoring the System Management Framework tool while the power loss to the computer occurs, therefore, a System Management Framework window will be displayed on the workstation/PC where the tester is located. (Test will be verified at each site.)

Input: Root map of System Management Framework window will be active in tester's display. Turn power off on computer.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the "faulty" computer (indicated by color of RED).

<u>Test Case 18</u> <u>Computer Monitoring Process Terminated</u>

This test verifies that the System Management Framework tool will detect and locate a computer "fault" when the monitoring process of the computer has been terminated. Testing includes monitoring the System Management Framework tool while the monitoring process is terminated, therefore, a System Management Framework window will be displayed on the workstation/PC where the tester is located. (Test will be verified at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Tester will locate and terminate/"kill" the process that monitors the computer.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the "faulty" computer (indicated by color or RED).

<u>Test Case 19</u> <u>Network Disconnect from Computer</u>

This test verifies that the System Management Framework tool will detect and locate a network disconnect from a computer terminal. Testing includes monitoring the System Management Framework tool while the network disconnect occurs, therefore, a System Management Framework window will be displayed on the workstation/PC where the tester is located. (Test will be verified at all sites.)

Input: Root map of System Management Framework window will be active in the tester's display. Disconnect network connect from computer terminal.

Output: Internet symbol on Framework has turned YELLOW/marginal status.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the "faulty" network interface (indicated by RED/critical status color) on the computer terminal where the network was disconnected.

The following series of tests verify that the System Management Framework tool, HP OpenView Network Node Manager, will properly detect and locate user environment faults consisting of operating system faults, hard disk capacity full, memory capacity full, excessive CPU load, and multi-process faults. In order to detect these faults, a MIB object will be defined within the System Management Framework tool to monitor each of the user environments. All fault events should be recorded in a problem log.

Test Case 20 Operating System Monitoring Process Terminated

This test verifies that the System Management Framework tool detects and locates the computer with the "failed" operating system. To simulate a failed operating system the monitoring process of the system will be terminated/"kill"ed, no processing will be harmed. Testing includes monitoring the System Management Framework tool while the monitoring process is terminated. (Test will be verified on all computers at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Opening an xterm window from a host machine, locate and "kill" the process that monitors the operating system.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the computer and actual operating system fault that occurred (Computer will be RED in color).

Test Case 21 Hard Disk Capacity Fault

This test verifies that the System Management Framework tool detects and locates the computer with a storage capacity fault. To simulate full capacity on a storage device, the tester will store large postscript or HDF files to the storage device. This test is limited to workstations and PCs with less than 1 GB of storage. (Test will be verified on all workstations/PCs at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Tester will store as many large postscript and HDF files to the system storage device as possible without damaging any existing files on the system.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the computer (indicated by color of RED) that contains the storage device fault. A message indicating storage capacity should also be displayed.

Test Case 22 Memory Threshold Fault

This test verifies that the System Management Framework tool detects and locates the computer with the memory threshold fault. Since most computers utilize a portion of their hard disk as virtual memory, the Management Information Base object created to monitor a computer's memory will contain threshold limits to determine when the memory capacity of a computer is in excessive use. (Test will be verified on all workstations/PCs at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Tester will execute an application that exceeds the memory threshold limit determined within the Management Information Base object created to monitor the computer memory usage. (If threshold limit is too high to exceed, lower the threshold as needed. After test, threshold limit will be returned to original state.)

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the computer (indicated by color of RED) that contains the memory threshold fault.

Test Case 23 CPU Threshold Fault

This test verifies that the System Management Framework tool detects and locates the computer where the CPU threshold limit has been exceeded. The Management Information Base object created will contain threshold limits to determine when the CPU load of a computer is in excessive use. (Test will be verified on all workstations/PCs at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Tester will execute an application that utilizes all/most of the CPU processing time.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the computer (indicated by color of RED) where the CPU threshold limit has been exceeded.

Test Case 24 Multi-Process Termination Fault

This test verifies that the System Management Framework tool detects and locates the computer and processes that have been abruptly terminated. In this test many processes will be activated (GUIs, ftps, etc..), then some of the processes will be abruptly 'killed' simulating software failures.

Input: Root map of System Management Framework window will be active in the tester's display. Activate many processes on a host machine. (FTPs, Telnets, GUIs, etc.) 'Kill' a quarter of the processes from the machine.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the computer and processes that were terminated.

4.1.4.2.3 Sequence 3 - System Network Statistics

The following series of tests verify that the System Management Framework tool is able to perform generation, collection, store and display of system network statistics at each local site. This series of tests will be run for each site supported by this release.

Test Case 1 Generation of Network Statistics

This test verifies that the System Management Framework tool is able to generate network statistics.

Input: System Management Framework window displayed on tester's machine.

Output: Log containing generated statistics.

Expected Results: Output GUI display of network statistics.

Test Case 2 Collection of Network Statistics

This test verifies that the System Management Framework tool is able to collect network statistics.

Input: System Management Framework window displayed on tester's machine.

Output: Log containing generated statistics.

Expected Results: Output log of network statistics.

Test Case 3 Storing Network Statistics

This test verifies that the System Management Framework tool collects and stores network statistics.

Input: System Management Framework window displayed on tester's machine.

Output: Log containing collected network statistics.

Expected Results: Local site statistics captured in event file.

Test Case 4 Display of Network Statistics

This test verifies that the System Management Framework tool accurately displays the network statistics gathered.

Input: System Management Framework window displayed on tester's machine.

Output: GUI display of generated statistics.

Expected Results: Local site statistics display capabilities.

4.2 Interim Release 1 System Threads

This section identifies and describes the functional system level threads which have been defined for Interim Release 1. Each thread contains a list of objectives, which describes the overall purpose of the thread, and a thread test description, which includes a list of dependencies, if applicable, without which the requirements allocated to the thread cannot be fully tested.

All Interim Release 1 requirements will be tested by verifying the functionality of the system level threads and builds. The mapping of the Interim Release 1 requirements to respective system level threads is summarized in Appendix A.

4.2.1 DAAC LAN Thread

The following listing identifies the test sequences within this thread, its paragraph reference number and page number of the accompanying text.

4.2.1.1 Thread Objectives

The DAAC LAN thread will provide the basic local area network communications necessary for the interface proof-of-concept delivered in Interim Release 1. Once established, these interfaces will provide later threads and builds with client/server capabilities. This thread will explicitly focus on the functionality, while consecutive threads that rely on the provided communication services, will implicitly test these capabilities each time a client/server transaction takes place.

4.2.1.2 Thread Test Description

The tester will be able to securely log in and out of the system both locally and remotely.

- Log on to a system. Tester should be provided with a prompt to enter a password. Enter tester-defined password to complete logon.
- Attempt to log on to a system. Enter an invalid account name. Verify that connectivity is refused.
- Attempt to log on to a system. Enter an invalid password for a valid account name. Verify that connectivity is refused.
- Log on to a system and enter valid account name and password. Logoff the system. Verify that connection to the system was closed.

DAAC LAN capabilities will provide the local sites (DAACs) with the ability to utilize the TCP/IP Suite along with the associated client/server functionality.

- Logon to a system and send a client a message. Verify the client received message. Have a client issue a message to your system. Verify message is received.

Demonstrate the ability to perform a remote login and telnet to another remote machine and perform a file transfer between two machines at the application level. The network level Internet services provide a connectionless packet delivery service, a reliable stream transport service, error and control messaging, and socket interfacing.

Utilizing COTS/public domain software, perform a file transfer (FTP, RCP, or DFS pipe) of a data file from one system to another. Verify the data file was properly transferred to the tester. Errors, if any, that may have been encountered during the transmission should be recorded in a status log. Review the status log for possible errors/transmission status.

Force an error to occur in the send of a file from one system to another. Review the status log to determine error condition/transmission status. Log file should indicate that the transmission failed and provide a reason for the failure.

Dependencies (if Applicable):

Test Support Requirements

- Hardware:
  - Bridges
  - Routers

- Software:
  - Multi-tester emulation capture and playback test tool
- Data:

4.2.1.2.1 Sequence 1 - System Logon and Logoff

The following tests verify proper security for logging on and off a system. These tests are to be repeated for each supported platform. Tests include using valid account names and passwords, invalid account names with valid passwords, valid account names and invalid passwords, invalid account names and invalid passwords, and valid and invalid logoffs to the system. For each case, an appropriate entry into the history log should be made.

Test Case 1 Logon - Valid Account Name and Password

This test verifies that a tester successfully logs on to a system after entering a valid account name and an associated password. The main menu/screen of the appropriate system is displayed. Furthermore, a successful logon entry is made in the history log.

Input: Valid account name and password.

Output: Main menu/screen displayed, successful logon entry in the history log.

Expected Results: The main menu/screen of the appropriate system is displayed to the tester and a successful logon entry is made in the history log.

Test Case 2 Logon - Invalid Account Name, Valid Password

This test verifies that connectivity is refused when an invalid account name is entered. This test is to be repeated for different types of invalid account names (i.e., null value, carriage returns, case-sensitive issues, less than the minimum number of characters for an account name, more than the maximum number of characters for an account name, etc.). A message is displayed indicating that an invalid account name was entered and returns the tester to the login prompt. Furthermore, an unsuccessful logon entry is made in the history log indicating that an invalid account name was entered.

Input: Invalid account name, valid password.

Output: Message displayed indicating invalid account name entered and sends tester back to login prompt, appropriate system log entry.

Expected Results: A message indicating that an invalid account name was entered is displayed and returns the tester to the login prompt. An unsuccessful logon entry in the history log is made indicating that an invalid account name was entered.

Test Case 3 Logon - Valid Account Name, Invalid Password

This test verifies that connectivity is refused when an invalid password is entered. This test is to be repeated for different types of invalid passwords (i.e., null value, carriage returns, case-sensitive issues, less than the minimum number of characters for a password, more than the maximum number of characters for a password, the word “password”, etc.). A message is displayed indicating that an incorrect password was entered and the tester is sent back to the login prompt. Furthermore, an unsuccessful logon entry in the history log is made indicating that an incorrect password was entered for that account.

Input:	Valid account name, invalid password.
Output:	Message displayed indicating an incorrect password entered and sends tester back to login prompt, appropriate system log entry.
Expected Results:	A message indicating that an incorrect password was entered and sends tester back to login prompt. An unsuccessful logon entry is made in the history log indicating that an incorrect password was entered for that account.

<u>Test Case 4</u> <u>Logons - All Valid</u>

This test verifies that accounts A through F are allowed access to the system when valid account names and associated passwords are entered at the logon prompt. A record of the successful logon is recorded in the history log file along with other related statistics, which is verified by the tester.

Input:	Valid account names/passwords for accounts A through F.
Output:	Main menu/screen is displayed to accounts A through F, records of the successful logons entered in the history log file.
Expected Results:	Connection to the system is established and the main menu/screen of the particular system is displayed to accounts A through F. Records of the logon are recorded in the history log file indicating the successful logon along with other related statistics.

<u>Test Case 5</u> <u>Logons - All Invalid</u>

This test verifies that accounts A through F are denied access to the system when invalid account names or invalid passwords are entered at the logon prompt. Invalid account names or passwords can be of the following: case-sensitive, carriage returns (i.e., “null” value), less or more than the minimum or maximum number of characters allowed for an account name or password, etc.. A message indicating the incorrect logon is displayed to the tester and the logon prompt is redisplayed to the tester. A record of the unsuccessful logon attempt is recorded in the history log file along with other related statistics, which is verified by the tester.

Input:	Invalid account names for accounts A through C. Valid account names and invalid passwords for accounts D through F.
Output:	Message indicating incorrect logon displayed, logon prompt redisplayed, records of the unsuccessful logon attempts.

Expected Results: Connection to the system is refused for accounts A through F and a message is displayed to the tester of an incorrect logon and redisplays the logon prompt. Each unsuccessful logon attempt is recorded in the history log file along with other associated data.

Test Case 6 Logons - Valid and Invalid

This test verifies that connection to the system is established for accounts A and B, who enter valid account names/passwords, and refused for accounts C through F, who enter either invalid account names or invalid passwords. Invalid account names or passwords can be of the following: case-sensitive issues, carriage returns (i.e., “null” value), less or more than the minimum or maximum number of characters allowed for an account name or password, etc.. In each case, the logon activity is recorded in the history log file along with other associated data.

Input: Valid account names/passwords for accounts A and B, invalid account names for accounts C and D, valid account names and invalid passwords for accounts E and F.

Output: Main menu/screen displayed to accounts A and B, message indicating incorrect logon displayed to accounts C through F, logon prompt redisplayed to accounts C through F, records of the successful or unsuccessful logons in the history log file.

Expected Results: Connection is established to accounts A and B and refused to accounts C through F. Statistics of the successful or unsuccessful logons are recorded in the history log file, which is verified by the tester.

Test Case 7 Valid Logoff

This test verifies that connectivity is properly closed after a tester (using account1) executes a valid logoff to the system. A tester using account2 monitors the activity of the system. The tester using account1 logs on to the system. Account2 verifies that activity starts for account1. The tester using account1 logs off the system and the tester using account2 verifies that account1 is no longer active on the system and that the port connection to account1 was successfully closed.

Input: account1 name/password, account2 name/password, logoff sequence from account1.

Output: Log file displayed to account2 showing the activity of the system before account1 logs on to the system, after account1 logs on to the system, and after account1 logs off the system.

Expected Results: Account2 monitors the activity of the system and verifies that activity starts for account1 when account1 logs on to the system and that activity stops when account1 logs off the system. Furthermore, the history log will record the logon and logoff to the system by each account.

Test Case 8 Invalid Logoff

This test verifies that connectivity is properly closed after an invalid logoff to the system has occurred. This test is to be repeated for different types of invalid logoffs (i.e., ctrl-c, UNIX "kill", turning computer off, etc.). A tester using account2 monitors the activity of the system. The tester using account1 logs on to the system. Account2 verifies that activity starts for account1. The tester using account1 executes an invalid logoff to the system and the tester using account2 verifies that account1 is no longer active on the system and that the port connection to account1 was successfully closed.

Input: account1 name/password, account2 name/password, invalid logoff sequence from account1.

Output: Log file displayed to account2 showing the activity of the system before account1 logs on to the system, after account1 logs on to the system, and after account1 is logged off the system.

Expected Results: Account2 monitors the activity of the system and verifies that activity starts for account1 when account1 logs on to the system and that activity stops when account1 is logged off the system. Furthermore, the history log will record the logon and logoff to the system by each account.

4.2.1.2.2 Sequence 2 - Communication on Same Host (E-mail)

The following series of tests verify that an account, via basic LAN capabilities, is able to communicate using e-mail with other accounts that are connected to the same host. These tests are to be repeated for each supported platform.

Test Case 1 Account2 Logged on System

This test verifies that a tester, with a valid account, is able to log on to a system and communicate, via e-mail, with another account, which is logged on to the same host. An e-mail message is sent from one account and then waits for an e-mail response from the other account. This verifies two-way communication capabilities.

Input: account1 name/password, account2 name/password, message1 (from account1), message2 (from account2).

Output: Message indicating account2 has mail. Message received by account2. Message indicating account1 has mail. Message received by account1. History log file records of all activity and transactions by the tester.

Expected Results: The history log will record the logon to the system by each account, it will record the transmission of the e-mail messages, it will record the resource usage, response time, and the number of transactions.

Test Case 2 Account2 Not Logged on System

This test verify that when account1 e-mails a message to account2 which is not logged on to the system, a message indicating account2 has mail is displayed when account2 logs on to the system.

Input: account1 name/password, e-mail message to account2.

Output: Message indicating account2 has mail. History log file records of all activity and transactions by the accounts.

Expected Results: The history log will record the logon to the system by each account, it will record the transmission of the e-mail message, it will record the resource usage, response time, and the number of transactions.

Test Case 3 Disconnect Account2 from Host

This test verifies that if account2 is disconnected from the host prior to receiving the e-mail message, then a message indicating account2 has mail is displayed when account2 reconnects to or logs back on to the system.

Input: account1 name/password, account2 name/password, e-mail message1 from account1, disconnect account2.

Output: Message indicating account2 has mail. History log file records of all activity and transactions by the accounts.

Expected Results: The history log will record the logon to the system by each account, it will record the transmission of the e-mail message, it will record the resource usage, response time, and the number of transactions.

4.2.1.2.3 Sequence 3 - Communication on Different Hosts

The following series of tests verifies that an account, via basic LAN capabilities, is able to communicate with other accounts, via e-mail, that are connected to different local hosts. These tests are to be repeated for each supported platform.

Test Case 1 Account2 Logged on Host 2

This test verifies that account1 is able to log on to a system (on Host 1) and communicate, via e-mail, with account2 which is logged on to another local host (on Host 2) via the Ethernet LAN. Message1 is sent from account1 and waits for the responding message2 from account2. This verifies two-way communication capabilities.

Input: account1 name/password, account2 name/password, e-mail message1 (from account1), e-mail message2 (from account2).

Output: Message indicating account2 has mail. Message received by account2. Message indicating account1 has mail. Message received by account1. History log file records of all activity and transactions by the tester.

Expected Results: The history log will record the logon to the system by each account, it will record the transmission of the e-mail messages, it will record the resource usage, response time, and the number of transactions.

Test Case 2 Account2 Not Logged on Host 2

This test verifies that if account1 (on Host 1) tries to e-mail a message to account2 which is not logged on to the local host (Host 2), the appropriate message indicating that account2 has mail is displayed when account 2 logs on to the system.

Input: account1 name/password, e-mail message sent to account2, account2 name/password.

Output: Message indicating account2 has mail. History log file records of all activity and transactions by the accounts.

Expected Results: The history log will record the logon to the system by each account, it will record the transmission of the e-mail message, it will record the resource usage, response time, and the number of transactions.

Test Case 3 Disconnect Account2 from Host 2

This test verifies that if account2 is disconnected from Host 2 prior to receiving the e-mail message from acccount1, then a message indicating account2 has mail is displayed when account2 reconnects to or logs back on to the system.

Input: account1 name/password (on Host 1), account2 name/password (on Host 2), e-mail message1 from account1, disconnect account2.

Output: Message indicating account2 has mail. History log file records of all activity and transactions by the accounts.

Expected Results: The history log will record the logon to the system by each account, it will record the transmission of the e-mail message, it will record the resource usage, response time, and the number of transactions.

4.2.1.2.4 Sequence 4 - FTP Data File Transfer

The following series of tests verify that a tester is able to perform FTP file transfers between two systems. Data types to be tested include unstructured text, binary unstructured, binary sequential, sequential text, and all combinations of these types. These tests are to be repeated for each supported platform.

Test Case 1 FTP Data File from Host 2 to Host 1 - Complete

This test verifies that a tester, using a valid account, is able to transmit a data file to an account on another local machine via ftp through the Ethernet LAN.

Input: account name/password on Host 1 and Host 2, data file for file transfer (ftp) from Host 2 to Host 1.

Output: ftp completes data file transfer from Host 2 to Host 1, Host 1 directory listings verify that data file was transferred.

Expected Results: The history log will record the logon onto the systems by the account, the transmission of the FTP transaction, the resource usage, response time, and the number of transactions. Checksum of data files on Host 2 and Host 1 should equal.

Test Case 2 FTP Data File from Host 1 to Host 2 - Complete

This test verifies that a tester, using a valid account, is able to transmit a data file to an account on another local machine via ftp through the Ethernet LAN.

Input: account name/password on Host 1 and Host 2, data file for file transfer (ftp) from Host 1 to Host 2.

Output: ftp completes data file transfer from Host 1 to Host 2, Host 2 directory listings verify that data file was transferred.

Expected Results: The history log will record the logon onto the systems by the account, the transmission of the FTP transaction, the resource usage, response time, and the number of transactions. Checksum of data files on Host 1 and Host 2 should equal.

Test Case 3 FTP Data File from Host 2 to Host 1 - Partial

This test verifies that the system properly handles the case when an error occurs during the transmission of a file from a local host via ftp through the Ethernet LAN. The tester forces an error in the transmission (either by exiting/quitting the application, a ctrl-c, or other means) to stop the transfer process before the transmission is complete.

Input: account name/password on Host 1 and Host 2, data file for file transfer (ftp) from Host 2 to Host 1.

Output: ftp completes partial data file transfer from Host 2 to Host 1, Host 1 directory listings verify that data file was transferred.

Expected Results: The history log will record the logon onto the systems by the account, the transmission of the FTP transaction, the resource usage, response time, and the number of transactions. Checksum of data files on Host 2 and Host 1 should not equal.

Test Case 4 FTP Data File from Host 1 to Host 2 - Partial

This test verifies that the system properly handles the case when an error occurs during the transmission of a file from a local host via ftp through the Ethernet LAN. The tester forces an error in the transmission (either by exiting/quitting the application, a ctrl-c, or other means) to stop the transfer process before the transmission is complete.

Input: account name/password on Host 1 and Host 2, data file for file transfer (ftp) from Host 1 to Host 2.

Output: ftp completes partial data file transfer from Host 1 to Host 2, Host 2 directory listings verify that data file was transferred.

Expected Results: The history log will record the logon onto the systems by the account, the transmission of the FTP transaction, the resource usage, response time, and the number of transactions. Checksum of data files on Host 1 and Host 2 should not equal.

4.2.1.2.5 Sequence 5 - RCP Data File Transfer

The following series of tests verify that a tester is able to perform data file transfers of various sizes via remote file copy (rcp) between two systems. Data types to be tested include unstructured text, binary unstructured, binary sequential, sequential text, and all combinations of these types. Tests include successful and unsuccessful transfers of files. These tests are to be repeated for each supported platform.

Test Case 1 RCP Data File from Host 2 to Host 1 - Complete

This test verifies that a tester, using a valid account, is able to transmit a data file to an account on another machine via remote file copy (rcp).

Input: account name/password on Host 1 and Host 2, data file for remote copy (rcp) from Host 2 to Host 1.

Output: rcp completes data file transfer from Host 2 to Host 1, Host 1 directory listings verify that data file was transferred.

Expected Results: The history log will record the logon onto the systems by account A, the transmission of the rcp transaction, the resource usage, response time, and the number of transactions. Checksum of data files on Host 2 and Host 1 should equal.

Test Case 2 RCP Data File from Host 1 to Host 2 - Complete

This test verifies that a tester, using a valid account, is able to transmit a data file to an account on another machine via remote file copy (rcp).

Input: account name/password on Host 1 and Host 2, data file for remote copy (rcp) from Host 1 to Host 2.

Output: rcp completes data file transfer from Host 1 to Host 2, Host 2 directory listings verify that data file was transferred.

Expected Results: The history log will record the logon onto the systems by account A, the transmission of the rcp transaction, the resource usage, response time, and the number of transactions. Checksum of data files on Host 1 and Host 2 should equal.

Test Case 3 RCP Data File from Host 2 to Host 1 - Partial

This test verifies that the system properly handles the case when an error occurs during the transmission of a file from a local host via remote file copy (rcp). The tester forces an error in the transmission (either by exiting/quitting the application, a ctrl-c, or other means) to stop the transfer process before the transmission is complete.

Input: account name/password on Host 1 and Host 2, data file for remote copy (rcp) from Host 2 to Host 1.

Output: rcp completes partial data transfer from Host 2 to Host 1, Host 1 directory listings verify that data file was transferred.

Expected Results: The history log will record the logon onto the systems by the account, the transmission of the rcp transaction, the resource usage, response time, and the number of transactions. Checksum of data files on Host 2 and Host 1 should not equal.

Test Case 4 RCP Data File from Host 1 to Host 2 - Partial

This test verifies that the system properly handles the case when an error occurs during the transmission of a file from a local host via remote file copy (rcp). The tester forces an error in the transmission (either by exiting/quitting the application, a ctrl-c, or other means) to stop the transfer process before the transmission is complete.

Input: account name/password on Host 1 and Host 2, data file for remote copy (rcp) from Host 1 to Host 2.

Output: rcp completes partial data file transfer from Host 1 to Host 2, Host 2 directory listings verify that data file was transferred.

Expected Results: The history log will record the logon onto the systems by the account, the transmission of the rcp transaction, the resource usage, response time, and the number of transactions. Checksum of data files on Host 1 and Host 2 should not equal.

4.2.1.2.6 Sequence 6 - Data File Transfer Through RPC Pipe

The following test verifies that a tester is able to perform data file transfers of various sizes using RPC Pipes. Data types to be tested include unstructured text, binary unstructured, binary sequential, sequential text, and all combinations of these types. Tests include successful and unsuccessful transfers of files. These tests are to be repeated for each supported platform.

Test Case 5 Transfer of File Using RPC Pipes

This test verifies that a tester, using a valid account, is able to transfer data files of various sizes using RPC Pipes.

Input: Valid login for account on Host 1 and Host 2 at the same site supported in IR-1. Data files for file transfer using RPC Pipes.

Output: Completed data file transfers from Host 1 to Host 2. Host 2 directory listings verify that the data files were transferred.

Expected Results: The history log will record the logon onto the systems, it will record the transmission of the RPC Pipe transactions, it will record the resource usage, response time, and the number of transactions. Checksum of data files on Host 1 and Host 2 should equal.

4.2.1.2.7 Sequence 7 - Data File Transfer Through DFS

The following test verifies that a tester is able to perform data file transfers of various sizes via DCE Distributed File Service. Data types to be tested include unstructured text, binary unstructured, binary sequential, sequential text, and all combinations of these types. Tests include successful and unsuccessful transfers of files. These tests are to be repeated for each supported platform.

<u>Test Case 6</u> <u>Transfer of File through Distributed File Service</u>

This test verifies that testers, using valid accounts with appropriate authenticity and authorization, are able to access data files of various sizes from any other host (within LAN or WAN) via DCE Distributed File Service. When given a filename, a DFS client cache manager queries the Cell Directory Service (CDS) for the address of a file set location server. The cache manager stores the address for subsequent use. The cache manager makes a remote procedure call to the file set location server to get the address of the file server serving the target file. The cache manager makes a remote procedure call (request to open file) to the file server. The file server verifies the client’s authenticity and, if the client is authorized, serves the file to the client.

Input: Valid login for accounts A, B, C, and D on local Host 1, valid login for accounts B and D on remote Host 2. Same filename request to Distributed File Service by all accounts. Only accounts A and B have appropriate authorization to transfer file.

Output: Accounts A and B receive data file requested on the respective host. Host directory listings verify that the data file was transferred. Message indicating lack of appropriate authorization to transfer file received by accounts C and D.

Expected Results: File is transferred into the local listings of the respective host for account’s A and B. A comparison of the local copy of the file to that of the remote copy should produce the same results. Accounts C and D receive message that the accounts did not have the appropriate authorization. The history log will record the logons and the file transmissions.

4.2.2 ESN WAN Thread

The following listing identifies the test sequences within this thread, its paragraph reference number and page number of the accompanying text.

4.2.2.1 Thread Objectives

The ESN WAN thread will provide the basic external network communications necessary for the interface proof-of-concept delivered in Interim Release 1. Once established, these interfaces will provide later threads and builds with client/server capabilities. This thread will explicitly focus on the functionality, while consecutive threads that rely on the provided communication services will implicitly test these capabilities each time a client/server transaction takes place.

4.2.2.2 Thread Test Description

ESN WAN capabilities will provide the external sites (SCFs, ADCs, DAACs, etc.) with the ability to utilize the TCP/IP suite along with the associated client/server functionality provided herein to open and close connections with the available DAACs for the purposes of data retrieval and toolkit updates.

- Perform successful remote interactive logons to GSFC, LaRC, and MSFC from EDF.
- Verify access to all machines at each site within the WAN.
- Verify V0 network between DAAC sites.
- Receive e-mail from a remote sites.
- Perform file transfers from the one site to another using ftp, rcp, DFS.

Dependencies (if Applicable):

Test Support Requirements

- Hardware:
- Software:
    - System Management Framework

4.2.2.2.1 Sequence 1 - Successful Remote Logons

The following series of tests verify connectivity from EDF to each DAAC site (GSFC, LaRC, MSFC). Testing includes successful logons within the EDF then accessing each site through the Internet.

Test Case 1 Remote Logon to GSFC from EDF.

This test verifies the LAN to WAN connectivity between the EDF and GSFC. The tester will log into the EDF LAN, and attempt a remote login to a host a GSFC. All activities will be recorded in the history log file.

Input: Valid login to EDF LAN. Valid login to host machine at GSFC.

Output: Successful logons to both EDF and GSFC hosts. History log file records logon activities.

Expected Results: Connection to the EDF LAN and GSFC through the WAN is established. Records of logon activities are recorded in the history log file indicating the successful logons along with other related statistics.

Test Case 2 Remote Logon to MSFC from EDF.

This test verifies the LAN to WAN connectivity between the EDF and MSFC. The tester will log into the EDF LAN, and attempt a remote login to a host a MSFC. All activities will be recorded in the history log file.

Input: Valid login to EDF LAN. Valid login to host machine at MSFC.

Output: Successful logons to both EDF and MSFC hosts. History log file records logon activities.

Expected Results: Connection to the EDF LAN and MSFC through the WAN is established. Records of logon activities are recorded in the history log file indicating the successful logons along with other related statistics.

Test Case 3 Remote Logon to LaRC from EDF.

This test verifies the LAN to WAN connectivity between the EDF and LaRC. The tester will log into the EDF LAN, and attempt a remote login to a host a LaRC. All activities will be recorded in the history log file.

Input: Valid login to EDF LAN. Valid login to host machine at LaRC.

Output: Successful logons to both EDF and LaRC hosts. History log file records logon activities.

Expected Results: Connection to the EDF LAN and LaRC through the WAN is established. Records of logon activities are recorded in the history log file indicating the successful logons along with other related statistics.

Test Case 4 Verification of History Log File

This test verifies that the history log file records all system accesses and activities. Manually, the tester will review the history log file to verify that tests 1-3 were completely recorded. The history log should contain correct information for the activities which occurred in this test sequence.

Input:	Test cases 1-3 of this sequence.
Output:	History log file print out.
Expected Results:	The history log should reflect all activities which occurred during this test sequence.

4.2.2.2.2 Sequence 2 - Verification of DAAC Access

The following series of tests verify that all host machines connected through the WAN at each DAAC site are accessible. Testing includes successful access to the EDF LAN then verifying the connectivity to each site.

<u>Test Case 1</u> <u>Login to EDF</u>

This test verifies that the tester is able to successfully logon to a host machine within the EDF.

Input:	Valid account name/password for tester on host within EDF.
Output:	Successful logon. History log file updated with tester activities.
Expected Results:	Connection to the system is established and the main screen of the host is displayed to the tester. The history log file is updated with the tester's logon activities.

<u>Test Case 2</u> <u>Access to GSFC</u>

This test verifies that all host machines connected to the LAN at GSFC are accessible through the network from the EDF.

Input:	Listing of all machines connected to GSFC LAN. "ping" all host machines on listing.
Output:	ping statistics displayed for each machine.
Expected Results:	All host machines listed from GSFC should return ping messages that are connected to GSFC LAN. History log will record all activities.

<u>Test Case 3</u> <u>Access to LaRC</u>

This test verifies that all host machines connected to the LAN at LaRC are accessible through the network from the EDF.

Input: Listing of all machines connected to LaRC LAN. "ping" all host machines on listing.

Output: ping statistics displayed for each machine.

Expected Results: All host machines listed from LaRC should return ping messages that are connected to LaRC LAN. History log will record all activities.

Test Case 4 Access to MSFC

This test verifies that all host machines connected to the LAN at MSFC are accessible through the network from the EDF.

Input: Listing of all machines connected to MSFC LAN. "ping" all host machines on listing.

Output: ping statistics displayed for each machine.

Expected Results: All host machines listed from MSFC should return ping messages that are connected to MSFC LAN. History log will record all activities.

Test Case 5 Verification of History Log File

This test verifies that the history log file records all system accesses and activities. Manually, the tester will review the history log file to verify that tests 1-4 were completely recorded. The history log should contain correct information for the activities which occurred in this test sequence.

Input: Test cases 1-4 of this sequence.

Output: History log file print out.

Expected Results: The history log should reflect all activities which occurred during this test sequence.

4.2.2.2.3 Sequence 3 - V0 Network Access

The following series of tests verify the V0 network connectivity used in IR-1. All tests will be performed on host machines within each site (GSFC, MSFC, LaRC, and EDF) that are directly connected to the V0 network. Since the V0 network is a dedicated network for data transfer only, the tester will "ping" dedicated host machines at the other sites from the dedicated host machine at GSFC.

Test Case 1 GSFC to EDF V0 Access

This test verifies that the dedicated V0 link from GSFC to EDF is operational. To perform this test the tester must logon to the dedicated V0 host machine within GSFC to test the "dedicated V0 link to EDF".

Input: Valid name/password for tester on dedicated V0 host machine at GSFC. 'ping' dedicated V0 host machine at EDF.

Output: Successful logon to V0 host machine. ping statistics from V0 host machine at EDF.

Expected Results: V0 host machine at EDF should return ping messages. History log file will record all test activities and ping statistics.

Test Case 2 GSFC to LaRC

This test verifies that the dedicated V0 link from GSFC to LaRC is operational. To perform this test the tester must logon to the dedicated V0 host machine within GSFC to test the "dedicated V0 link to LaRC".

Input: Valid name/password for tester on dedicated V0 host machine at GSFC. 'ping' dedicated V0 host machine at LaRC.

Output: Successful logon to V0 host machine. ping statistics from V0 host machine at LaRC.

Expected Results: V0 host machine at LaRC should return ping messages. History log file will record all test activities and ping statistics.

Test Case 3 GSFC to MSFC

This test verifies that the dedicated V0 link from GSFC to MSFC is operational. To perform this test the tester must logon to the dedicated V0 host machine within GSFC to test the V0 link to MSFC.

Input: Valid name/password for tester on dedicated V0 host machine at GSFC. 'ping' dedicated V0 host machine at GSFC.

Output: Successful logon to V0 host machine. ping statistics from V0 host machine at MSFC.

Expected Results: V0 host machine at MSFC should return ping messages. History log file will record all test activities and ping statistics.

Test Case 4 GSFC to EDC

This test verifies that the dedicated V0 link from GSFC to EDC is operational. To perform this test the tester must logon to the dedicated V0 host machine within GSFC to test the V0 link to EDC.

Input: Valid name/password for tester on dedicated V0 host machine at GSFC. 'ping' dedicated V0 host machine at GSFC.

Output: Successful logon to V0 host machine. ping statistics from V0 host machine at EDC.

Expected Results: V0 host machine at EDC should return ping messages. History log file will record all test activities and ping statistics.

Test Case 5 Verification of History Log File

This test verifies that the history log file records all system accesses and activities. Manually, the tester will review the history log file to verify that tests 1-3 were completely recorded. The history log should contain correct information for the activities which occurred in this test sequence.

Input: Test cases 1-3 of this sequence.

Output: History log file print out.

Expected Results: The history log should reflect all activities which occurred during this test sequence.

4.2.2.2.4 Sequence 4 - Remote File Transfer

The following series of tests demonstrates that a valid account with the proper system access is able to transfer a file from one site (DAAC, SCF, ADC, etc.) to another. Within this series the tester will verify the ability to transfer data files from DAAC to DAAC.

Test Case 1 Transmit File from EDF to GSFC (ftp)

This test verifies that a tester, using a valid account, is able to transmit a file from EDF to an account on host machine at GSFC. This test verifies the connectivity of the EDF LAN to the system WAN.

Input: Valid login for tester on Host 1 at EDF and Host 2 at GSFC. Data file for file transfer (ftp) from Host 1 to Host 2.

Output: ftp completes data file transfer from Host 1 to Host 2. Host 2 directory listings verify that data file was transferred.

Expected Results: The history log file will record the logon to both hosts, it will record the transmission of the ftp transaction, it will record the resource usage, response time, and the number of transactions. Checksum of data files on Host 1 and Host 2 should equal.

Test Case 2 Transmit File from EDF to LaRC (ftp)

This test verifies that a tester, using a valid account, is able to transmit a file from EDF to an account on host machine at LaRC. This test verifies the connectivity of the EDF LAN to the system WAN.

Input: Valid login for tester on Host 1 at EDF and Host 2 at LaRC. Data file for file transfer (ftp) from Host 1 to Host 2.

Output: ftp completes data file transfer from Host 1 to Host 2. Host 2 directory listings verify that data file was transferred.

Expected Results: The history log file will record the logon to both hosts, it will record the transmission of the ftp transaction, it will record the resource usage, response time, and the number of transactions. Checksum of data file on Host 1 and Host 2 should equal.

Test Case 3 Transmit File from EDF to MSFC (ftp)

This test verifies that a tester, using a valid account, is able to transmit a file from EDF to an account on host machine at MSFC. This test verifies the connectivity of the EDF LAN to the system WAN.

Input: Valid login for tester on Host 1 at EDF and Host 2 at MSFC. Data file for file transfer (ftp) from Host 1 to Host 2.

Output: ftp completes data file transfer from Host 1 to Host 2. Host 2 directory listings verify that data file was transferred.

Expected Results: The history log file will record the logon to both hosts, it will record the transmission of the ftp transaction, it will record the resource usage, response time, and the number of transactions. Checksum of data file on Host 1 and Host 2 should equal.

Test Case 4 Transmit File from EDF to EDC (ftp)

This test verifies that a tester, using a valid account, is able to transmit a file from EDF to an account on host machine at EDC. This test verifies the connectivity of the EDF LAN to the system WAN.

Input: Valid login for tester on Host 1 at EDF and Host 2 at EDC. Data file for file transfer (ftp) from Host 1 to Host 2.

Output: ftp completes data file transfer from Host 1 to Host 2. Host 2 directory listings verify that data file was transferred.

Expected Results: The history log file will record the logon to both hosts, it will record the transmission of the ftp transaction, it will record the resource usage, response time, and the number of transactions. Checksum of data file on Host 1 and Host 2 should equal.

Test Case 5 Transmit File from EDF to GSFC (rcp)

This test verifies that a tester, using a valid account, is able to transmit a data file to an account on a machine at GSFC via remote file copy (rcp).

Input: Valid login for tester on Host 1 at EDF and Host 2 at GSFC. Data file for remote file copy (rcp) from Host 1 to Host 2.

Output: rcp completes data file transfer from Host 1 to Host 2. Host 2 directory listings verify that data file was transferred.

Expected Results: The history log file record the logon onto both hosts, it will record the transmission of the rcp transaction, it will record the resource usage, response time, and the number of transactions. Checksum of data files on Host 1 and Host 2 should equal.

Test Case 6 Transmit File from EDF to LaRC (rcp)

This test verifies that a tester, using a valid account, is able to transmit a data file to an account on a machine at LaRC via remote file copy (rcp).

Input: Valid login for tester on Host 1 at EDF and Host 2 at LaRC. Data file for remote file copy (rcp) from Host 1 to Host 2.

Output: rcp completes data file transfer from Host 1 to Host 2. Host 2 directory listings verify that data file was transferred.

Expected Results: The history log file record the logon onto both hosts, it will record the transmission of the rcp transaction, it will record the resource usage, response time, and the number of transactions. Checksum of data files on Host 1 and Host 2 should equal.

Test Case 7 Transmit File from EDF to MSFC (rcp)

This test verifies that a tester, using a valid account, is able to transmit a data file to an account on a machine at MSFC via remote file copy (rcp).

Input: Valid login for tester on Host 1 at EDF and Host 2 at MSFC. Data file for remote file copy (rcp) from Host 1 to Host 2.

Output: rcp completes data file transfer from Host 1 to Host 2. Host 2 directory listings verify that data file was transferred.

Expected Results: The history log file record the logon onto both hosts, it will record the transmission of the rcp transaction, it will record the resource usage, response time, and the number of transactions. Checksum of data files on Host 1 and Host 2 should equal.

Test Case 8 Transmit File from EDF to EDC (rcp)

This test verifies that a tester, using a valid account, is able to transmit a data file to an account on a machine at EDC via remote file copy (rcp).

Input: Valid login for tester on Host 1 at EDF and Host 2 at EDC. Data file for remote file copy (rcp) from Host 1 to Host 2.

Output: rcp completes data file transfer from Host 1 to Host 2. Host 2 directory listings verify that data file was transferred.

Expected Results: The history log file record the logon onto both hosts, it will record the transmission of the rcp transaction, it will record the resource usage, response time, and the number of transactions. Checksum of data files on Host 1 and Host 2 should equal.

Test Case 9 Transfer of File Through Distributed File Service

This test verifies that a tester, using a valid account, is able to access a data file from another host, whether local or remote, via DCE Distributed File Service.

Input: Valid login for tester on Host 1. Data file wanted.

Output: Tester receives data file requested.

Expected Results: Given tester's account authenticity and authorization, file is transferred through DCE Distributed File Service into tester's local listings. The history log file will record the login and the file transmission.

Test Case 10 Verification of History Log File

This test verifies that the history log file records all system accesses and activities. Manually, the tester will review the history log file to verify that tests 1-7 were completely recorded. The history log should contain correct information for the activities which occurred in this test sequence.

Input: Test cases 1-7 of this sequence.

Output: History log file print out.

Expected Results: The history log should reflect all activities which occurred during this test sequence.

4.2.2.2.5 Sequence 5 - Communications

The following series of tests verify that testers are able to communicate to remote systems, via basic LAN and WAN capabilities. This test also verifies that basic communication services are available. All activities will be recorded in the history log file.

Test Case 1 E-mail from EDF to GSFC

This test verifies that a tester using valid accounts, via basic LAN and WAN capabilities, is able to send e-mail messages from an account at EDF to an account at GSFC.

Input: Valid account names/passwords for accounts at both EDF and GSFC. Tester sends e-mail message from EDF account at GSFC.

Output: Connection to the respective hosts, message received by account at GSFC.

Expected Results: Tester receives e-mail message from EDF at GSFC. History log file will record all activities and transactions.

Test Case 2 E-mail from EDF to LaRC

This test verifies that a tester using valid accounts, via basic LAN and WAN capabilities, is able to send e-mail messages from an account at EDF to an account at LaRC.

Input: Valid account names/passwords for accounts at both EDF and LaRC. Tester sends e-mail message from EDF account at LaRC.

Output: Connection to the respective hosts, message received by account at LaRC.

Expected Results: Tester receives e-mail message from EDF at LaRC. History log file will record all activities and transactions.

Test Case 3 E-mail from EDF to MSFC

This test verifies that a tester using valid accounts, via basic LAN and WAN capabilities, is able to send e-mail messages from an account at EDF to an account at MSFC.

Input: Valid account names/passwords for accounts at both EDF and MSFC. Tester sends e-mail message from EDF account at MSFC.

Output: Connection to the respective hosts, message received by account at MSFC.

Expected Results: Tester receives e-mail message from EDF at MSFC. History log file will record all activities and transactions.

Test Case 4 E-mail from EDF to EDC

This test verifies that a tester using valid accounts, via basic LAN and WAN capabilities, is able to send e-mail messages from an account at EDF to an account at EDC.

Input: Valid account names/passwords for accounts at both EDF and EDC. Tester sends e-mail message from EDF account at EDC.

Output: Connection to the respective hosts, message received by account at EDC.

Expected Results: Tester receives e-mail message from EDF at EDC. History log

Test Case 5 E-mail from DAAC to DAAC

This test verifies that a tester, using valid accounts, is able to send e-mail between DAAC sites. This test will be completed at each DAAC site.

Input: Valid account names/passwords for accounts at both DAACs. Tester sends e-mail message from site to site.

Output: Connection to the respective hosts, message received by account at DAAC 2.

Expected Results: Tester receives e-mail message from DAAC 1 at DAAC 2. History log file will record all activities and transactions.

Test Case 6 Verification of History Log File

This test verifies that the history log file records all system accesses and activities. Manually, the tester will review the history log file to verify that tests 1-3 were completely recorded. The history log should contain correct information for the activities which occurred in this test sequence.

Input: Test cases 1-3 of this sequence.

Output: History log file print out.

Expected Results: The history log should reflect all activities which occurred during this test sequence.

Test Case 7 Fault Notification sent via NSI

This test verifies that the NSI, NASA Space Internet, transmitting a fault notification from GSFC to EDF. The fault notification will be in the form of a consistently formatted electronic message that can be automatically parsed by a receiving program from ECS. It will contain enough information to determine the nature of the fault and which sites are affected.

Input: The NSI schedules preventive maintenance to one of its connections to GSFC. The fault notification will contain the preventive maintenance schedule as to when the maintenance begins, ends, and to what locations are affected. In this case, the EDF is affected.

Output: Fault notification sent electronically as an alert. Also, the fault notification contributes to an audit trail that assist with performing network analysis.

Expected Results: The fault notification will reflect all of the pertinent information users need to know.

Test Case 8 TCP/IP connection from EDF to GSFC

This test verifies EDF can access GSFC via TCP/IP connect.

Input: A tester simulated at EDF, logs onto a workstation at EDF. After the tester gets the login prompt, the tester rlogins into the GSFC network.

Output: The tester connects to the GSFC network and the history log file captures this activity.

Expected Results: The history log should reflect all activities which occurred during this test sequence. The tester connects to the GSFC network.

4.2.3 TRMM TSDIS Interface Thread

The following listing identifies the test sequences within this thread, its paragraph reference number and page number of the accompanying text.

4.2.3.1 Thread Objectives

The objectives of this thread are to verify the data exchange capabilities between the TRMM Science Data and Information System (TSDIS) and the ECS EDF, GSFC DAAC and MSFC DAAC, respectively.

4.2.3.2 Thread Test Description

The thread tests sequentially verify the capability of the TSDIS to interface and exchange data and information with the EDF, the GSFC DAAC and the MSFC DAAC. EDF and DAAC specific data sets and data exchanges with the TSDIS are used in each test.

Dependencies: (If Applicable)

- System Access Build

Test Support Requirements

- Hardware:

  - Workstation (mid-size)

- Software:

  - Xwindows software

  - Automated GUI Test Tool

  - HP OpenView

  - Data Comparison Tools

  - ClearCase

- Data:

  - Selected V0 data (HDF and native formatted)

  - Representative Science Algorithms written in C and FORTRAN

  - Selected VIRS, PR, TMI and Ground Validation (GV) data sets, or simulated data sets

  - Ancillary and correlative data or simulated ancillary and correlative data

4.2.3.2.1 Sequence 1 - Data Ingest, Handling and Data Transfer

The following series of tests verify the capabilities of the ECS to ingest data from the TSDIS; of the GSFC DAAC and the MSFC DAAC to receive product delivery schedule information and simulated data products from the TSDIS; of the ECS to send ancillary and correlative data to the TSDIS; and of the ECS to be capable of requesting reprocessing support from the TSDIS.

<u>Test Case 1</u> <u>TSDIS Data Product Delivery Schedule Receipt</u>

This test verifies the capability of the GSFC DAAC and the MSFC DAAC to be capable of receiving from the TSDIS information about data product delivery schedules.

Input: Real or simulated set of data product delivery schedules, in TSDIS specified format; set of corrupted or incorrect data to verify rejection of bad data.

Output: Receipt of transmitted data sets by the GSFC and MSFC DAACs identical to those sent.

Expected Results: Interpretation and acceptance by the two DAACs of the TSDIS transmitted product delivery schedule messages for correct data; rejection with error message for incorrect data.

<u>Test Case 2</u> <u>GSFC DAAC VIRS Data Product Receipt</u>

This test verifies the capability of the GSFC DAAC to receive data products from the TSDIS. The data products received from the TSDIS include Visible Infrared Scanner (VIRS) L1A through L3 processed data sets; associated metadata and browse data; directory and catalog information; and related algorithms and documentation. The GSFC DAAC shall have the capability to archive these data and be capable of retrieving these data upon request.

Input: Real or simulated sets of VIRS data. Sequential transmission, receipt and interpretation by the GSFC DAAC of VIRS L1A through L3 data sets; of VIRS metadata and browse data; of directory and catalog information; and of VIRS scientific algorithms and related documentation text files will be performed during these tests.

Output: Received VIRS data and information files, in accordance with data and information sets transmitted to the GSFC DAAC.

Expected Results: Concurrence that the GSFC DAAC has properly received and interpreted each of the data and information types described above.

<u>Test Case 3</u> <u>MSFC DAAC Data Product Receipt</u>

This test verifies the capability of the MSFC DAAC to receive data products from the TSDIS. The data products received from the TSDIS include Precipitation Radar (PR), TRMM Microwave Imager (TMI) and Ground Validation (GV) L1A through L3 processed data sets; associated metadata and browse data; directory and catalog information; and related algorithms and documentation. The MSFC DAAC shall have the capability to archive these data and be capable of retrieving these data upon request.

Input: Real or simulated sets of PR, TMI and GV data. Sequential transmission, receipt and interpretation by the MSFC DAAC of PR, TMI and GV L1A through L3 data sets; of PR, TMI and GV metadata and browse data; of directory and catalog information; of PR and TMI scientific algorithms; and of related documentation text files will be verified during these tests.

Output: Received PR, TMI and GV data and information files, in accordance with data and information sets transmitted to the GSFC DAAC.

Expected Results: Concurrence that the MSFC DAAC has properly received and interpreted each of the data and information types described above.

Test Case 4 ECS Transmission of Correlative and Ancillary Data to the TSDIS

This test verifies the capability of the ECS to retrieve from archive and to transmit to the TSDIS correlative and ancillary data, as requested and required by the TSDIS to perform its higher level data processing.

Input: Real or simulated sets of correlative and ancillary data, perhaps from the NOAA ADC or the V0 system, of compatible format to that required by the TSDIS.

Output: Receipt of transmitted data sets by the TSDIS identical to those sent.

Expected Results: Concurrence by the TSDIS that the transmitted data messages were correctly received and of correct format and content.

Test Case 5 Data Reprocessing Request Transmissions to the TSDIS

The following series of tests verify the capability of the TSDIS to receive from the GSFC DAAC and from the MSFC DAAC L1A-L3 and ancillary data sets to support TSDIS data product reprocessing. The test verifies the capability of the GSFC DAAC and the MSFC DAAC to transmit to the TSDIS requests for reprocessing of data to Level 1A through Level 3. The data reprocessing requests will include any needed correlative and ancillary data required by TSDIS to perform its reprocessing.

Input: Simulated L1A-L3 data sets and associated ancillary data sets, in proper format for receipt by the TSDIS; corrupted or incorrect data sets to test that the TSDIS will reject incorrect data.

Output: Receipt of transmitted data sets by the TSDIS identical to those sent by the GSFC and MSFC DAAC.

Expected Results: Interpretation and acceptance by the TSDIS of the transmitted data products and concurrence of the correctness in format and type to support TSDIS data reprocessing. Receipt from the TSDIS of acknowledgment of receipt of a correct data reprocessing request. Receipt from the TSDIS of error message upon TSDIS receipt of incorrect data.

4.2.3.2.2 Sequence 2 - ECS Testability and Test Support Capability

The following sequence verifies the capability of the ECS to support ECS testing in all phases of IR-1 development and verifies the testability of the ECS in the IR-1 time frame. A combination of test, inspection, analysis and demonstration is used to verify these capabilities.

In addition the capability of the ECS to be able to port relevant capabilities among the different sites is verified by demonstration and analysis. The transparent portability of hardware, software and interfaces among DAAC sites is verified.

Test Case 1 ECS Testability

This test verifies the capability to support ECS testing in all phases of IR-1 development, from unit to segment to system. In addition to actual test results and capability demonstrations, the verification of ECS's testability and of its ability to support testing will be performed by inspection of test plans and test equipment, as well as analysis of unit, segment and system test capabilities and test environments.

Input: Ongoing ECS test and demonstration results; information and specification of ECS test environments; test plans; analyses of test requirements; and specification of ECS testability criteria.

Output: Test results and test demonstrations; analyses of test environments and test plans; analysis of ECS testability criteria.

Expected Results: Confirmation of the testability of the ECS and of its capabilities to support unit, segment and system testing.

Test Case 2 ECS Portability

This test verifies the ECS portability among the different DAAC sites. The portability of ECS hardware and software capabilities will be verified by a combination of demonstration and analysis.

Input: A specification of the different hardware environments at each DAAC and a summary of the capabilities which need to be ported among the DAACs. Specification of DAAC hardware, support software and related documentation. Specification of different site and environment capabilities and interface requirements.

Output: A series of demonstration results showing identical ECS performance and ECS capabilities at different sites; a series of analyses of hardware and software specifications for different ECS hardware and support software systems.

Expected Results: Verification of the portability of the ECS capabilities among the different DAAC sites, to the extent required to support IR-1 requirements and capabilities.

4.2.4 TRMM SDPF Interface Thread

The following listing identifies the test sequences within this thread, its paragraph reference number and page number of the accompanying text.

4.2.4.1 Thread Objectives

The objectives of this thread are to verify the TRMM data exchange capabilities between the GSFC Sensor data processing facility (SDPF) the MSFC DAAC, the LaRC DAAC and the ECS EDF respectively.

4.2.4.2 Thread Test Description

The thread tests verify the capability of the SDPF to interface and exchange data and information with the ECS EDF, the MSFC DAAC, and the LaRC DAAC. EDF and DAAC specific data sets and data exchanges with the SDPF are used in each test.

Dependencies: (If Applicable)

- System Access Build

Test Support Requirements

- Hardware:
  - Workstation (mid-size)
- Software:
  - Xwindows software
  - Automated GUI Test Tool
  - HP OpenView
  - SCF and DAAC versions of PGS Toolkit
  - Data Comparison Tools
  - POSIX Checkers
  - ANSI certified compilers/linkers
- Data:
  - TRMM instrument CERES and LIS simulated Level 0 data products
  - Representative Science Algorithms written in C and FORTRAN
  - Calibration coefficient files
  - Ancillary data or simulated ancillary data

4.2.4.2.1 Sequence 1 - External Interface Readiness

The following series of tests verify the availability of the external interfaces. The various interface locations and interaction vary as well as the expected information being passed.

Test Case 1 TRMM LIS Data from SDPF to MSFC DAAC

This test verifies the interface between SDPF and MSFC for Level 0 data products and re-processing request activities.

Input: Real or simulated set of TRMM LIS data.

Output: Receipt of transmitted data at MSFC DAAC are identical to those sent.

Expected Results: Confirmation by the SDPF of receipt of acceptable data.

Test Case 2 TRMM LIS Corrupted Data from SDPF to MSFC DAAC

This test verifies that the system gives an error message when corrupted TRMM LIS data is sent to MSFC DAAC.

Input: Corrupted or incorrect, real or simulated set of TRMM LIS data in SDPF specified format.

Output: System gives an error message.

Expected Results: Rejection with error message for incorrect data.

Test Case 3 TRMM CERES Data from SDPF to LaRC DAAC

This test verifies the interface between SDPF and LaRC for Level 0 data products and re-processing request activities.

Input: Real or simulated set of TRMM CERES data in SDPF specified format.

Output: Receipt of transmitted data sets at LaRC DAAC are identical to those sent.

Expected Results: Confirmation by the SDPF of receipt of acceptable data.

Test Case 4 TRMM CERES Corrupted Data from SDPF to LaRC DAAC

This test verifies that the system gives an error message when corrupted TRMM CERES data is sent to MSFC DAAC.

Input: Corrupted or incorrect, real or simulated set of TRMM CERES data in SDPF specified format;

Output: System gives an error message.

Expected Results: Rejection with error message for incorrect data.

Test Case 5 TRMM SDPF Schedules from SDPF to EDF

This test verifies the interface between SDPF and EDF for status information and schedule adjudication of data exchange with DAACs.

Input: Real or simulated set of product delivery schedules, in SDPF specified format;

Output: Receipt of transmitted schedules are identical to those sent.

Expected Results: Confirmation by the SDPF of receipt of acceptable schedules.

Test Case 6 TRMM SDPF Corrupted Schedules from SDPF to EDF

This test verifies that the system gives an error message when corrupted TRMM schedule data is sent to DAACs.

Input: Corrupted or incorrect, real or simulated set of TRMM SDPF delivery schedule in SDPF specified format.

Output: System gives an error message.

Expected Result: Rejection with error message for incorrect delivery schedule.

Test Case 7 TRMM SDPF Predictive Orbit data from SDPF to MSFC DAAC

This test verifies that the interface between SDPF and MSFC for predictive orbit data products and re-processing request activities.

Input: Real or simulated set of TRMM predictive orbit data in SDPF specified format.

Output: Receipt of transmitted predictive orbit data sets are identical to those sent.

Expected Result: Confirmation by the SDPF of receipt of acceptable predictive orbit data.

Test Case 8 TRMM SDPF Corrupted Predictive Orbit data from SDPF to MSFC DAAC

This test verifies that the system gives an error message when corrupted TRMM SDPF predictive orbit data is sent to MSFC DAAC.

Input: Corrupted or incorrect, real or simulated set of TRMM predictive orbit data in SDPF specified format.

Output: System gives an error message.

Expected Result: Rejection with error message for incorrect data.

Test Case 9 TRMM SDPF Predictive Orbit Data from SDPF to LaRC DAAC

This test verifies that the interface between SDPF and LaRC DAAC for predictive orbit data products and re-processing request activities.

Input: Real or simulated set of TRMM predictive orbit data in SDPF specified format.

Output: Data received at LaRC DAAC Receipt of transmitted data sets are identical to those sent.

Expected Result: Confirmation by the SDPF of receipt of acceptable data.

Test Case 10 TRMM SDPF Corrupted Predictive Orbit Data from SDPF to LaRC DAAC

This test verifies that the system gives the error message when corrupted TRMM SDPF predictive orbit data is sent to LaRC DAAC.

Input: Corrupted or incorrect data, real or simulated set of TRMM predictive orbit data in SDPF specified format.

Output: System gives an error message.

Expected Result Rejection with error message for incorrect data.

Test Case 11 TRMM SDPF Definitive Orbit Data from SDPF to MSFC DAAC

This test verifies that the interface between SDPF and MSFC DAAC for definitive orbit data products and re-processing request activities.

Input: Real or simulated set of TRMM definitive orbit data in SDPF specified format.

Output: Data received at MSFC DAAC Receipt of transmitted data sets are identical to those sent.

Expected Result: Confirmation by the SDPF of receipt of acceptable data.

Test Case 12 TRMM SDPF Corrupted Definitive Orbit Data from SDPF to MSFC DAAC

This test verifies that the system gives the error message when corrupted TRMM SDPF data is sent to MSFC DAAC.

Input: Corrupted or incorrect data, real or simulated set of TRMM definitive orbit data in SDPF specified format.

Output: System gives an error message.

Expected Result: Rejection with error message for incorrect data.

Test Case 13 TRMM SDPF Definitive Orbit Data from SDPF to LaRC DAAC

This test verifies that the interface between SDPF and LaRC DAAC for definitive orbit data products and re-processing request activities.

Input: Real or simulated set of TRMM definitive orbit data in SDPF specified format.

Output: Data received at LaRC DAAC Receipt of transmitted data sets are identical to those sent.

Expected Result: Confirmation by the SDPF of receipt of acceptable data.

Test Case 14 TRMM SDPF Corrupted Definitive Orbit Data from SDPF to LaRC DAAC

This test verifies that the system gives an error message when corrupted TRMM SDPF definitive orbit data is sent to LaRC DAAC.

Input: Corrupted or incorrect, real or simulated set of TRMM definitive orbit data in SDPF specified format.

Output: System gives an error message.

Expected Result: Rejection with error message for incorrect data.

4.2.5 Algorithm I&T Preparatory Thread

The following listing identifies the test sequences within this thread, its paragraph reference number and page number of the accompanying text.

4.2.5.1 Thread Objectives

The objective of this thread is to demonstrate the ability to maintain configuration control of the IR-1 science algorithms, and PGS Toolkits utilizing the CM tool ClearCase as well as demonstrating that other tools provided can be used to transport an algorithm from an SCF to DAAC environment. The tools provided to support the transition are:

- SCF and DAAC version of PGS tool kits
- ANSI certified compilers/linkers
- Data Comparison tool
- POSIX checkers.

The PGS Toolkits provide an isolation layer between the PGS and the Science Algorithm. This isolation prevents changes in the PGS from impacting the science algorithm. The SCF version of the toolkit provides a development environment that emulates critical DAAC PGS functions. In the DAAC version of the PGS Toolkit the emulated capabilities are removed and real PGS functionality is integrated.

4.2.5.2 Thread Test Description

Selected Science System algorithms, source code for the PGS Toolkits and IR-1 system software will be stored in the CM software libraries at each site. Storage of the software will be verified by:

- reviewing/inspecting the CM libraries
- performing UNIX directory listings
- physical review of hard copy materials

Executable code for the applicable platforms for each site will be:

- built
- checksum
- verified
- installed on applicable platforms
- COTS packages will be installed on all applicable platforms

Science Algorithms will be received and configured into CM prior to performing any compliance checking, compiling, or linking. Algorithm resident under CM can also be checked out at users requests. Science Algorithms may contain all or some of the following :

- source code
- associated documentation
- calibration coefficients

- test procedures
- test data

The ability to check in GFE, COTS and Public Domain database files into CM also are verified. Science software developed in the SCF must operate successfully in a DAAC operational environment. Tests are conducted on SCF developed software in the I&T environment to demonstrate proper operation in the DAAC environments. Testing includes access to subroutines and libraries to determine if SCF software can port to a DAAC site. Algorithms are tested for proper operation in the DAAC environments.

Portability of science algorithm are tested by comparison toolkits. Confirmation of attaining ECS software standards are done by ANSI certified compiler and POSIX compliance checkers.

Dependencies (if Applicable):

Test Support Requirements

- Hardware:
  - Simulated DAAC and SCF facilities
  - ECS defined platforms
- Software:
  - CM Tool
  - PGS Toolkits (DAAC and SCF versions)
  - COTS Packages
    - Portable Operating System Interface For Computer Environment (POSIX) Checkers
    - ANSI certified compilers/linkers
    - Data Comparison Tools
  - Representative science algorithms written in C and FORTRAN
- Data:
  - The test data will include the following categories of files:
    - Instrument science data (CERES and LIS)
    - Ancillary data
    - Calibration coefficient files

4.2.5.2.1 Sequence 1 - Building/Installing the CM Platform

The following test verifies the capability to build the CM platform. This entails installing the CM Tool and the source code/libraries for the PGS Toolkits, and COTS packages on the designated CM platform.

Test Case 1 Build and Install CM Platform

This test verifies the ability to establish a workstation (platform) as the repository for the CM Tool and all the associated source code, libraries, and executable for algorithms, and COTS packages.

Input: CM Tool Software package, source code, libraries, and executables for science algorithms, and COTS packages. Checksum values for all delivered source code, libraries, executables, etc. Network environment to support file transfers or tape I/O capability.

Output: An established CM repository, appropriate status/error messages and log entries generated as a result of building/installing the CM workstation.

Expected Results: All checksums for source code, libraries, executables etc. should match those provided by CM.

4.2.5.2.2 Sequence 2 - Building Executable Codes for Each Platform Type

The following test verifies the ability to build executable code for ECS defined platform type that EOSDIS is required to support. The number of platform types may be limited for the release.

Test Case 1 Building Executable Code for an ECS Defined Platform

This test verifies the ability to build executable code for an ECS defined platform.

Input: Installed CM repository workstation with source code, libraries, and make files to support creation of executable codes for the ECS defined platform.

Output: Executable code for the ECS defined platform, appropriate status/error messages and log entries generated as a result of executing make files.

Expected Results: Valid executable codes ready to be installed on the ECS defined platform. Executable checksum values for one ECS defined platform are not expected to match those of a different ECS defined platform type.

4.2.5.2.3 Sequence 3 - Installing/Verifying Executable Codes for ECS Defined Platform Type

The following test verifies the ability to install the executable code for ECS defined platform type. ECS will be required to support installation of executable codes for the platform types that are available. In addition, this test will demonstrate the ability to verify the installation of the executable on ECS defined platform type.

Test Case 1 Installing/Verifying Executable codes for the ECS Defined Platform Type

This test demonstrates the ability to install and verify the installation of the executable code and COTS software packages for the ECS defined platform type.

Input: Executable codes and their checksum values for the ECS defined platform type. Installation procedures/scripts required to support the platform install. COTS packages that must be installed. Network environment to support file transfers or Tape I/O capability.

Output: Appropriate status/error messages and log entries generated as a result of software installation.

Expected Results: An ECS defined platform capable of performing the functions/operations provided as part of the IR-1 system release.

4.2.5.2.4 Sequence 4 - Build/Install Executable Code for Previous and Updated Software Versions

The following series of tests demonstrate the ability to generate an executable code for a previous or new (updated) version of the software baseline and install it on the required platform. Testing will be performed using three cases: (1) the previous version of an executable code is always retained in the CM area. To restore an older executable would require installing the old executable code, (2) the previous versions of the executable code are not resident in CM and must be rebuilt and installed on the required platform, and (3) modifications have been performed to existing source code which will result in a new (updated) version of a executable code. The executable code must be built and installed.

Test Case 1 Install Previous Version of an Executable Code

This test demonstrates the ability to install a previous version (other than the current version number) of an executable code on a required platform.

Input: All (or some) of the previous versions of executable code and their checksum values for the platform type located on the CM repository. Network environment to support file transfers or tape I/O capability.

Output: Appropriate status/error messages and log entries generated as a result of installing the previous version of the executable.

Expected Results: Tester was able to find and select a previous version of a executable code using the CM tool, install it on the required platform and verify that it will execute on the platform after installation.

Test Case 2 Build/Install Previous Version of an Executable Code

This test demonstrates the ability to build and install a previous version (other than the current version number) of an executable code on a required platform.

Input: All of the previous versions of source code and libraries required to build an older version of an executable code, the configuration record used to build the previous binary and the platform type the binary is being built for. Network environment to support file transfers or Tape I/O capability to support installation of the executable code.

Output: Appropriate status/error messages and log entries generated as a result of building and installing the previous version of the executable code.

Expected Results: Tester was able to find and select the configuration record used to build the previous version of the executable code using the CM tool, build the executable code, install it on the required platform, and verify that it will execute on the platform after installation.

Test Case 3 Build/Install New (Updated) Version of an Executable Code

This test demonstrates the ability to build and install an updated (modified) version of a executable code on a required platform.

Input: The updated version of source code, libraries, the makefile required to build the updated (new) version of a executable code, and the platform type the executable code is being built for. Network environment to support file transfers or Tape I/O capability to support installation of the executable code.

Output: Appropriate status/error messages and log entries generated as a result of building and installing the updated (new) version of the executable code.

Expected Results: Tester was able to build the updated executable code using the CM tool, install it on the required platform, and verify that it will execute on the platform after installation.

4.2.5.2.5 Sequence 5 - Check-in/Check-out Algorithms to/from CM

The following series of tests demonstrate the systems ability to receive and check into Configuration Management any algorithms provided by the science community. In addition, users must be able to "check-out" any algorithm resident under CM Control, modify it and then check it back into CM.

Test Case 1 Process (Check-in) Algorithm to Configuration Management

This test verifies the ability to receive algorithms by the science community and bring them under Configuration Management Control under the designated CM Tool.

Input: Science Algorithms and all associated data to include source code, author, benchmark test procedures/data/results, and compiler version/identification. Designated CM repository and CM Tool. Network environment to support file transfer or Tape I/O capability.

Output: Appropriate status/error messages and log entries generated as a result of loading science algorithm into CM using the CM Tool.

Expected Results: Version 1 of the Science Algorithm and all its associated data should be resident on the designated CM repository and accessible by all valid users.

Test Case 2 Check-Out Algorithm from Configuration Management To User

This test verifies the ability to check an algorithm under CM Control out to the science community or other users.

Input: Designated CM repository and CM Tool. Algorithms resident under CM Control. Network environment and valid users that can "check-out" an algorithm from CM.

Output: Appropriate status/error messages and log entries generated as a result of checking science algorithm out from CM using the CM Tool.

Expected Results: Version logged out will reflect "checked out" to "xxx user".

Test Case 3 Check-Out Single Algorithm from Configuration Management To Multiple Users

This test verifies the ability to check a single algorithm under CM Control out to multiple users of the science community.

Input: Designated CM repository and CM Tool. Algorithms resident under CM Control. Network environment and valid users that can "check-out" an algorithm from CM.

Output: Appropriate status/error messages and log entries generated as a result of checking science algorithm out from CM using the CM Tool.

Expected Results: Version logged out will reflect "checked out" to "xxx user". Multiple users should be able to check out the same version of an algorithm while only the first user to "check-out" the algorithm will have the "reserved' copy.

Test Case 4 Process (Check-in) Modified Algorithms to Configuration Management

This test verifies the ability to receive algorithms modified by the science community and check them into Configuration Management as an updated version using the designated CM Tool.

Input: Original version for the modified Science Algorithm must already be resident in CM. Modified science algorithms and any updates to its supporting data (test files/documentation, etc.). CM repository and CM Tool. Network environment to support file transfer or tape I/O capability.

Output: Appropriate status/error messages and log entries generated as a result of loading modified science algorithm into CM using the CM Tool. Updated Version of science algorithm and any associated data resident on the designated CM repository. Original algorithm "checked-in" should not have changed.

Expected Results: Original Version 1 of the Science Algorithm plus updated version (Version 1 plus DIFF to make up new version of the algorithm) resident under CM and accessible by all valid users.

<u>Test Case 5</u> <u>Check-in of Single Algorithm with Multiple "Check-Outs"</u>

This test verifies the ability to maintain Configuration Management Control of system algorithms when one algorithm is checked out by more than a single user simultaneously.

Input: Original version of the Science Algorithm must already be resident in CM. CM repository and CM Tool. Multiple valid users to "check-out" the same Science Algorithm. Network environment to support file transfer or TAPE I/O capability.

Output: Appropriate status/error messages and log entries generated as a result of multiple users checking the same algorithm out from CM utilizing the CM Tool. Appropriate status/error messages and log entries generated as a result of multiple users checking the same algorithm back into CM utilizing the CM Tool.

Expected Results: Algorithm correctly reflects "checked-out" when an authorized user has obtained proper access to the algorithm through the CM Tool. First user to obtain copy of the algorithm will have the "reserved" copy. Any user that is NOT the first user to "check-out" the algorithm from CM should NOT be able to check the algorithm back into CM while the reserved copy remains "checked-out".

<u>Test Case 6</u> <u>Enhanced Algorithms Under Configuration Management at the EDF</u>

This test demonstrates that the SMC provides overall management of CMed enhanced Algorithms by retaining a master copy at the EDF.

Input: Original version of the science algorithm must be present in CM repository. Request is made to CM for enhanced version of the algorithm.

Output: Appropriate status/error message and log entries occurs when a request is made and a file is ready to be transferred to the requester. File is transferred by ftp from EDF to requester. An acceptance code is received from requester verifying the status of the delivery.

Expected Results: SMC sends the enhanced version of the algorithm and maintains original version and the enhanced version in the CM library at the EDF.

4.2.5.2.6 Sequence 6 - Check-in GFE, and Public Domain Data Bases to CM

The following test demonstrates the ability to receive GFE, COTS, and Public Domain database files and check them into CM to establish Configuration Control of the files.

Test Case 1 Check-In GFE, COTS, and/or Public Domain Database Files To CM

This test verifies the ability to check GFE, COTS, and/or Public Domain Databases into CM.

Input: Designated CM repository and CM Tool. GFE, COTS, or Public Domain database files to be "checked-in" to CM. Network environment to support file transfers or tape I/O capability.

Output: Appropriate status/error messages and log entries generated as a result of checking the database file into CM using the CM Tool.

Expected Results: Version 1 of the GFE, COTS, and/or Public Domain database should be resident on the designated CM repository and be accessible by all valid users.

4.2.5.2.7 Sequence 7 - Toolkit Data Standards/Compliance Tests

The following series of tests evaluates ECS standards checking tools. Tests are conducted in the integration and test environment representing DAAC operational environments to confirm compliance with ECS standards. Tests are conducted in a test environment which is independent but identical to various site operational production environments.

Test Case 1 ANSI Certified Compilers/linkers Test.

This test runs the makefiles and executes scripts for the compilation, loading and execution of science software provided by SCF. This is done for software developed in C, Ada and

FORTRAN programming languages. Testing includes running algorithms which were successfully run at the SCF. Algorithms are tested on all approved DAAC platforms.

Input: Science software developed at the SCFs. Access to DAAC compilers and linkers. At least one algorithm developed in C, FORTRAN will be run. Algorithms without errors are used for input.

Output: Compiled code and status report.

Expected Results: Code compiled and linked at the DAACs is compared to code compiled and linked at the SCF. No differences should be found between SCF compiled code and DAAC compiled code.

Test Case 2 ANSI Certified Compilers/linkers Error Test.

This test runs the makefiles and executes scripts for the compilation, loading and execution of science software provided by SCF. Testing includes running algorithms which contain known errors. This is done for software developed in C and FORTRAN programming languages. Algorithms are tested on all approved DAAC platforms.

Input: Science software developed at the SCFs. Access to DAAC compilers and linkers. At least one algorithm developed in C, FORTRAN will be run. Algorithms containing known errors are used for input.

Output: Status and error reporting.

Expected Results: Algorithms containing errors should display complete and appropriate error messages.

Test Case 3 POSIX Compliance Test

This test involves running science software algorithms developed at a SCF against a POSIX checker to determine the software's ability to operate in the DAAC environment. An algorithm verified as having no known errors is run against the checker. This is done for algorithms in written in C and FORTRAN on all DAAC platforms.

Input: Science software developed at the SCFs. Access to the POSIX compliance checker. At least one algorithm developed in C and FORTRAN will be run on platforms representing all DAAC environments. Algorithms without errors are used for input.

Output: Status messages.

Expected Results: Valid algorithms will return a status of compliance.

Test Case 4 POSIX Non - Compliance Test

This test involves running science software algorithms developed at a SCF against a POSIX checker to determine the software's ability to operate in the DAAC environment.

An algorithm with known errors is tested (by rerunning the verified algorithm after changing POSIX API calls to non POSIX API calls) to confirm the checker's ability to detect POSIX non-compliance. This is done for algorithms in written in C and FORTRAN on all DAAC platforms.

Input: Science software developed at the SCFs. Access to the POSIX compliance checker. At least one algorithm developed in C and FORTRAN will be run on platforms representing all DAAC environments. Algorithms without errors and algorithms containing known errors are used for input.

Output: Status messages and error reporting

Expected Results: Invalid algorithms will return status of non-compliance. If non-compliance is found, all errors inserted in the representative algorithms should be detected.

Test Case 5 Standards Enforcement Test

In this test SCF science software, including algorithms and calibration coefficients are run against ECS standards checking software. EOSDIS standards checkers, which enforce ECS compliance, are run against the SCF developed algorithms. Standards checkers include code checkers, and any other tools which confirm compliance to ECS standards. SCF software is tested for completeness and correct format.

Input: Science software developed at the SCFs. Access to standards tools. Algorithms without errors are used for input.

Output: Status messages and error reporting

Expected Results: Status is displayed. No error messages are displayed.

Test Case 6 Standards Enforcement Error Test

In this test SCF science software which is in violation of ECS standards, including algorithms and calibration coefficients, are run against ECS standards checking software. Standards checkers include code checkers, and any other tools which confirm compliance to ECS standards.

Input: Science software developed at the SCFs modified to include violations to ECS standards. Access to standards tools. Algorithms containing known errors are used for input.

Output: Status messages and error reporting

Expected Results: Notification is displayed on the terminal screen of status and errors. All errors should be detected and status should state the error with enough detail to understand the nature of the non-compliance to ECS standards.

4.2.5.2.8 Sequence 8 - Toolkit Portability Tests

The following series of tests assess whether SCF developed algorithms successfully transfer to a DAAC environment. The use of comparison tools will confirm that algorithm results obtain from running the algorithms in the DAAC environment are the same as results obtained from running the algorithms in the SCF environment.

Test Case 1 Data Comparison Tool for Identical Algorithms

This test is to determine if a comparison tool adequately compares two identical algorithms. An algorithm is run in the SCF environment and deemed accurate. This algorithm is run in the DAAC environment. The results are compared using the algorithm comparison tool.

Input: An algorithm (and algorithm output) successfully run in the SCF environment. Access to data comparison tools. This algorithm is run in DAAC environments. The results from running the algorithm are used to test the comparison tool.

Output: Status messages and error reporting from the comparison tool.

Expected Results: The tool should indicate no differences between algorithm results.

Test Case 2 Data Comparison Tool for Differing Algorithms

This test determines if a comparison tool adequately detects inconsistencies in output from two algorithms with known differences. An algorithm is run in the SCF and deemed valid. These same algorithm is modified and run again in the SCF. A comparison tool is used to compare the algorithm and the algorithm after modification. All outputs from the algorithms and modifications made to the algorithm are recorded. The SCF algorithm prior to modification and the SCF algorithm after modification is then run in a DAAC environment. Again the comparison tool is used to compare the algorithm results. Finally the output from the comparison tool used in the SCF is compared to the output from the comparison tool used in the DAAC.

Input: An algorithm and modified algorithm (and algorithm output) run in the SCF environment. Access to data comparison tools. This algorithms are run in DAAC environments. The results from running the algorithms are used to test the comparison tool.

Output: Algorithm output. Status and error reporting from the comparison tool.

Expected Results: The tool should indicate differences based upon modifications made to the algorithm. The results from the comparison tool used at the SCF should be the same as the results obtained from the comparison tool used at the DAAC. SCF scientists will work with DAAC personnel to determine acceptable comparison tool output if differences are found.

Test Case 3 Data Comparison Tools for all ECS Defined Platforms

This test verifies that a data comparison tool will capture all differences between an algorithm run on different platforms. The same algorithm is run on two or more different platforms. Using a comparison tool the algorithm are compared. Differences are analyzed to determine if the results given by the comparison tool are complete and accurate.

Input: A valid algorithm is run on different platforms in the SCF. A comparison tool is used to compare results of the same algorithm run on different platform in the SCF. The results are recorded. The same algorithm is run on different platforms in the DAAC. Again a comparison tool is used to compare the results of the algorithm run on different DAAC platforms. The DAAC results are recorded.

Output: The results from the comparison tool run at the SCF are compared to the results recorded at the DAAC.

Expected Results: There should little to no difference between SCF comparison tool output and DAAC comparison tool output. SCF scientists will work with DAAC personnel to determine acceptable comparison tool output if differences are found.

4.2.5.2.9 Sequence 9 - Algorithm/Method Toolkit Tests

The following series of tests assess algorithms that utilize PGS Toolkits. Algorithms developed and accepted as valid by the SCF, using the SCF Toolkit in the SCF environment, are run in the DAAC environment. The algorithm/method is tested using the data sets (or references) delivered with the software. Test results are analyzed to validate the correctness of the data products.

Test Case 1 Geolocation/Geocoordination Conversion Test

This test verifies that an algorithm developed and deemed valid at the SCF, will run correctly giving the same algorithm output as produced in the SCF when run at the DAAC. Representative SCF algorithms that utilizes PGS Toolkit functions for geolocation/geocoordinate transformations are run on all approved DAAC platforms. The results are analyzed to determine the scientific correctness of the data products.

Input: Algorithms developed at the SCF and SCF /DAAC Toolkits. Access to all subroutines and libraries containing data needed to perform data conversions. These algorithms will use the PGS Toolkit for geolocation/geocoordinate transformations. The algorithm is first run at the DAAC using the PGS SCF Toolkit version. Then the algorithm is run at the DAAC using the DAAC Toolkit version.

Output: Algorithm status, algorithm product output for both the SCF Toolkit and the DAAC Toolkit.

Expected Results: The product outputs from the SCF and DAAC Toolkits are analyzed for correct scientific product content. There should be similar output from the SCF and DAAC Toolkits. Outputs from the SCF and

DAAC Toolkits may be examined using a proven data comparison tool.

Test Case 2 Time/Date Conversion Test

This test verifies that an algorithm developed and deemed valid at the SCF, will run correctly giving the same algorithm output as produced in the SCF when run at the DAAC. Representative SCF algorithms that utilizes PGS Toolkit functions for time/date conversions are run on all approved DAAC platforms. The results are analyzed to determine the scientific correctness of the data products.

Input: Algorithms developed at the SCF and SCF /DAAC Toolkits. Access to all subroutines and libraries containing data needed to perform data conversions. These algorithms will use the PGS Toolkit for time/date conversions. The algorithm is first run at the DAAC using the PGS SCF Toolkit version. Then the algorithm is run at the DAAC using the DAAC Toolkit version.

Output: Algorithm status, algorithm product output for both the SCF Toolkit and the DAAC Toolkit.

Expected Results: The product outputs from the SCF and DAAC Toolkits are analyzed for correct scientific product content. There should be similar output from the SCF and DAAC Toolkits. Outputs from the SCF and DAAC Toolkits may be examined using a proven data comparison tool.

Test Case 3 Calibration Coefficients and Algorithm Update Test

This test verifies the DAAC's ability to receive and update science software using new or modified (updated) algorithms/calibration coefficients. This test will evaluate the ability of the DAAC Toolkit environment to modified existing science software when an algorithm/coefficient update is deemed necessary by the SCF. An algorithm proven to be a valid algorithm from a previous test (Geolocation/Geocoordination Conversion Test or Time/Date Conversion Test) is modified and run at the SCF. The results are recorded. The same algorithm is modified at the DAAC using updated procedures and data received from the SCF. The results of the SCF output are compared to the DAAC output.

Input: Algorithms validated in previous tests such as the Geolocation/Geocoordination Conversion Test or Time/Date Conversion Test. New or updated algorithms and calibration coefficients. Access to all subroutines and libraries containing data needed to perform data conversions. Access to a script editor to make changes to the algorithms.

Output: Algorithm status, algorithm product output for both the SCF Toolkit and the DAAC Toolkit.

Expected Results: The product outputs from the SCF and DAAC Toolkits are analyzed for correct scientific product content. There should be similar output from the SCF and DAAC Toolkits. Outputs from the SCF and DAAC Toolkits may be examined using a proven data comparison tool.

4.2.6 ECS Administrative Thread

The following listing identifies the test sequences within this thread, its paragraph reference number and page number of the accompanying text.

4.2.6.1 Thread Objectives

The objectives of this thread are to verify the administrative capabilities of ECS which includes system management, system administration and system operations.

4.2.6.2 Thread Test Description

The ECS Administrative thread sequentially verifies the capability of system management, establishes management policies and procedures, system network statistics, office automation information and maintenance & operation for system administration.

Dependencies: (If Applicable)

Test Support Requirements

- Hardware:

  - Workstation (mid-size)

- Software:

  - Xwindows software
  - Automated GUI Test Tool
  - HP OpenView
  - SCF and DAAC versions of PGS Toolkit
  - Data Comparison Tools
  - ClearCase
  - POSIX Checkers

- ANSI certified compilers/linkers

- Data:
  - TRMM instrument CERES and LIS simulated Level 0 data products
  - Representative Science Algorithms written in C and FORTRAN
  - Calibration coefficient files
  - Ancillary data or simulated ancillary data

4.2.6.2.1 Sequence 1 - Monitoring of Hardware/Software/User Environment

The following series of tests verify that the System Management Framework tool will properly detect all hardware components active on the system at all times. These tests will confirm that all hardware, specific to the system, at GSFC, MSFC, LaRC, and the EDF are on-line and accessible through the System Management Framework tool. Testing will include contacting the DAAC liaisons or System Administrators and receiving a list of all hardware (PCs, workstations, minis, main-frames, archives, gateways/routers, printers, modems, etc.) currently active and in-use at each site.

<u>Test Case 1</u> <u>GSFC Hardware Confirmation</u>

This test verifies that the System Management Framework tool properly detects and monitors all hardware at Goddard Space Flight Center.

Input: GSFC submap of System Management Framework tool is active in the tester's display. Contact DAAC liaison and receive a list of all hardware components active and in-use.

Output: GSFC submap of System Management Framework tool displays all hardware active at GSFC. Listing of all hardware active and in-use at GSFC.

Expected Results: Hardware displayed in GSFC submap of System Management Framework tool matches/confirms hardware that is obtained from the DAAC liaison.

<u>Test Case 2</u> <u>MSFC Hardware Confirmation</u>

This test verifies that the System Management Framework tool properly detects and monitors all hardware at Marshall Space Flight Center.

Input: MSFC submap of System Management Framework tool is active in the tester's display. Contact DAAC liaison and receive a list of all hardware components active and in-use.

Output: MSFC submap of System Management Framework tool displays all hardware active at MSFC. Listing of all hardware active and in-use at MSFC.

Expected Results: Hardware displayed in MSFC submap of System Management Framework tool matches/confirms hardware that is obtained from the DAAC liaison.

Test Case 3 LaRC Hardware Confirmation

This test verifies that the System Management Framework tool properly detects and monitors all hardware at Langley Research Center.

Input: LaRC submap of System Management Framework tool is active in the tester's display. Contact DAAC liaison and receive a list of all hardware components active and in-use.

Output: LaRC submap of System Management Framework tool displays all hardware active at LaRC. Listing of all hardware active and in-use at LaRC.

Expected Results: Hardware displayed in LaRC submap of System Management Framework tool matches/confirms hardware that is obtained from the DAAC liaison.

Test Case 4 EDF Hardware Confirmation

This test verifies that the System Management Framework tool properly detects and monitors all hardware at the ECS Development Facility.

Input: EDF submap of System Management Framework tool is active in the tester's display. Contact System Administrator and receive a list of all hardware components active and in-use.

Output: EDF submap of System Management Framework tool displays all hardware active at EDF. Listing of all hardware active and in-use at EDF.

Expected Results: Hardware displayed in EDF submap of System Management Framework tool matches/confirms hardware that is obtained from the System Administrator.

The following series of tests evaluates the System Management Framework Tool, HP OpenView Network Node Manager, will properly detect and locate hardware faults (peripherals, archives, printer, etc.) that are connected within the LAN/WAN network. Hardware faults may be caused by power loss, terminating the process that monitors the hardware, or a network disconnect from the hardware. All fault events should be recorded in a problem log.

Test Case 5 Hardware Power Loss

This test verifies that the System Management Framework tool will detect a hardware fault caused from a power loss. Testing includes monitoring the System Management Framework tool while the power loss to the hardware occurs. A System Management Framework window will be displayed on the workstation/PC where the tester is located. (Test will be verified at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Turn power off on hardware (peripherals, archives, etc.).

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the inactive piece of hardware (indicated by color of RED).

Test Case 6 Hardware Monitoring Process Terminated

This test verifies that the System Management Framework tool will detect and locate a hardware "fault" when the monitoring process of the specific piece of hardware has been terminated. Testing includes monitoring the System Management Framework tool while the monitoring process is terminated, therefore, a System Management Framework window will be displayed on the workstation/PC where the tester is located. (Test will be verified at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Tester will locate and terminate/"kill" the process that monitors the hardware.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the "faulty" piece of hardware (indicated by color RED).

Test 7 Network Disconnect from Hardware (Peripherals, Printers, Archives, etc.)

This test verifies that the System Management Framework tool will detect and locate a network disconnect from a piece of hardware. Testing includes monitoring the System Management Framework tool while the network disconnect occurs, therefore, a System Management Framework window will be displayed on the workstation/PC where the tester is located. (Test will be verified at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Disconnect network connect from hardware.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW/marginal status.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the "faulty" network interface (indicated by RED/critical status color) on the piece of hardware where the network was initially disconnected.

The following series of tests verify that the System Management Framework Tool, HP OpenView Network Node Manager, will properly detect and locate network faults caused by gateway/router outages. Gateway/Router outages may be caused by power loss or by terminating the process that monitors the gateway/router.

<u>Test Case 8</u> <u>Gateway/Router Power Loss</u>

This test verifies that the System Management Framework tool detects and locates a network fault caused from a gateway/router power loss. Testing includes monitoring the System Management Framework tool while the power loss occurs.

A System Management Framework window will be displayed on the workstation where the tester is located. (Test will be verified at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Turn power off on gateway/router.

Output: Internet symbol on Root map of System Management Framework has turned RED.

Expected Results: Traversing through the Internet submaps, following the RED/critical status symbols, the tester should be directed to the faulty gateway/router (indicated by color of RED).

<u>Test Case 9</u> <u>Gateway/Router Monitoring Process Terminated</u>

This test verifies that the System Management Framework tool detects and locates a possible fault with a gateway/router. Testing includes displaying the System Management Framework tool, while the tester terminates the process that monitors the gateway/router activities. (Test will be verified at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Tester will locate and "kill" the process that monitors the gateway/router.

Output: Internet symbol on Root map of System Management Framework has turned RED.

Expected Results:	Traversing through the Internet submaps, following the RED/critical status symbols, the tester should be directed to the faulty gateway/router (color should be RED).

Test Case 10 Network Disconnect from Gateway/Router

This test verifies that the System Management Framework tool detects and locates a network fault caused from a network disconnect to the gateway/router. Testing includes monitoring the System Management Framework tool while the network disconnect occurs. A System Management Framework window will be displayed on the workstation where the tester is located. (Test will be verified at each site.)

Input:	Root map of System Management Framework window will be active in tester's display. Disconnect network connection from gateway/router.
Output:	Internet symbol on Root map of System Management Framework has turned RED.
Expected Results:	Traversing through the Internet submaps, following the RED/critical status symbols, the tester should be directed to the network interface of the gateway/router. (This interface should be RED/critical.)

The following series of tests verify that the System Management Framework tool, HP OpenView Network Node Manager, will properly detect and identify the software applications that have been abnormally terminated. Software application termination may be caused by terminating the actual software application itself or by terminating the monitoring process from the System Management Framework tool. All fault events should be recorded in a problem log.

Test Case 11 Software Application Termination

This test verifies that the System Management Framework tool will properly detect and locate the software application that was terminated abnormally. Testing includes monitoring the System Management Framework tool while the abnormal termination of the software application occurs, therefore, a System Management Framework window will be displayed on the workstation/PC where the tester is located. (Test will be verified at each site.)

Input:	Root map of System Management Framework window will be active in the tester's display. Tester will locate and terminate/"kill" the process that controls the software application.
Output:	Internet symbol on Root map of System Management Framework tool has turned YELLOW.
Expected Results:	Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the test should be directed to the computer where the application was running (computer will be RED in color) and the application itself.

Test Case 12 Software Application Monitoring Process Termination

This test verifies that the System Management Framework tool detects and locates the terminated software application when the monitoring process of the application has been terminated. Testing includes monitoring the System Management Framework tool while the monitoring process is terminated, therefore, a management window will be displayed on the workstation/PC where the tester is located. (Test will be verified at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Tester will locate and terminate/"kill" the process that monitors the software application.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the computer where the application was running (computer will be RED in color) and the application itself.

The following tests verify that the MIB objects created within the System Management Framework tool to detect and monitor all software processes (GUIs, OSs, FTPs, etc.) active on the system at all times. These tests will confirm that all software processes, specific to the system, at GSFC, MSFC, LaRC, and the EDF are monitored through the System Management Framework tool. Testing will include contacting the DAAC liaisons or System Administrators and receiving a list of all software applications and processes currently active at each site.

Test Case 13 GSFC Software Process Monitoring

This test verifies that the MIB created within the System Management Framework tool properly monitors all software processes at Goddard Space Flight Center.

Input: GSFC submap of System Management Framework tool is active in the tester's display. Contact DAAC liaison and receive a list of all software processes that are active and should be monitored.

Output: GSFC submap of System Management Framework tool displays all software processes active at GSFC. Listing of all software processes active at GSFC.

Expected Results: Software processes displayed in GSFC submap of System Management Framework tool matches/confirms software processes listed from DAAC liaison.

Test Case 14 MSFC Software Process Monitoring

This test verifies that the MIB created within the System Management Framework tool properly monitors all software processes at Marshall Space Flight Center.

Input: MSFC submap of System Management Framework tool is active in the tester's display. Contact DAAC liaison and receive a list of all software processes that are active and should be monitored.

Output: MSFC submap of System Management Framework tool displays all software processes active at MSFC. Listing of all software processes active at MSFC.

Expected Results: Software processes displayed in MSFC submap of System Management Framework tool matches/confirms software processes listed from DAAC liaison.

Test Case 15 LaRC Software Process Monitoring

This test verifies that the MIB created within the System Management Framework tool properly monitors all software processes at Langley Research Center.

Input: LaRC submap of System Management Framework tool is active in the tester's display. Contact DAAC liaison and receive a list of all software processes that are active and should be monitored.

Output: LaRC submap of System Management Framework tool displays all software processes active at LaRC. Listing of all software processes active at LaRC.

Expected Results: Software processes displayed in LaRC submap of System Management Framework tool matches/confirms software processes listed from DAAC liaison.

Test Case 16 EDF Software Process Monitoring

This test verifies that the MIB created within the System Management Framework tool properly monitors all software processes at the ECS Development Facilities.

Input: EDF submap of System Management Framework tool is active in the tester's display. Contact System Administrator and receive a list of all software processes that are active and should be monitored.

Output: EDF submap of System Management Framework tool displays all software processes active at EDF. Listing of all software processes active at EDF.

Expected Results: Software processes displayed in EDF submap of System Management Framework tool matches/confirms software processes listed from System Administrator.

The following series of tests verify that the System Management Framework Tool, HP OpenView Network Node Manager, will properly detect and locate network faults caused by computer (PCs, workstations, mini's, main frames) outages. Computer outages may be caused by power loss, terminating monitoring process, or network disconnect from the computer itself. All fault events should be recorded in a problem log.

Test Case 17 Computer Power Loss

This test verifies that the System Management Framework tool detects and locates a network fault caused from a computer power loss. Testing includes monitoring the System Management Framework tool while the power loss to the computer occurs, therefore, a System Management Framework window will be displayed on the workstation/PC where the tester is located. (Test will be verified at each site.)

Input:	Root map of System Management Framework window will be active in tester's display. Turn power off on computer.
Output:	Internet symbol on Root map of System Management Framework has turned YELLOW.
Expected Results:	Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the "faulty" computer (indicated by color of RED).

Test Case 18 Computer Monitoring Process Terminated

This test verifies that the System Management Framework tool will detect and locate a computer "fault" when the monitoring process of the computer has been terminated. Testing includes monitoring the System Management Framework tool while the monitoring process is terminated, therefore, a System Management Framework window will be displayed on the workstation/PC where the tester is located. (Test will be verified at each site.)

Input:	Root map of System Management Framework window will be active in the tester's display. Tester will locate and terminate/"kill" the process that monitors the computer.
Output:	Internet symbol on Root map of System Management Framework has turned YELLOW.
Expected Results:	Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the "faulty" computer (indicated by color or RED).

Test Case 19 Network Disconnect from Computer

This test verifies that the System Management Framework tool will detect and locate a network disconnect from a computer terminal. Testing includes monitoring the System Management Framework tool while the network disconnect occurs, therefore, a System Management Framework window will be displayed on the workstation/PC where the tester is located. (Test will be verified at all sites.)

Input:	Root map of System Management Framework window will be active in the tester's display. Disconnect network connect from computer terminal.

Output: Internet symbol on Framework has turned YELLOW/marginal status.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the "faulty" network interface (indicated by RED/critical status color) on the computer terminal where the network was disconnected.

The following series of tests verify that the System Management Framework tool, HP OpenView Network Node Manager, will properly detect and locate user environment faults consisting of operating system faults, hard disk capacity full, memory capacity full, excessive CPU load, and multi-process faults. In order to detect these faults, a Management Information Base (MIB) object will be defined within the System Management Framework tool to monitor each of the user environments. All fault events should be recorded in a problem log.

Test Case 20 Operating System Monitoring Process Terminated

This test verifies that the System Management Framework tool detects and locates the computer with the "failed" operating system. To simulate a failed operating system the monitoring process of the system will be terminated/"kill"ed, no processing will be harmed. Testing includes monitoring the System Management Framework tool while the monitoring process is terminated. (Test will be verified on all computers at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Opening an xterm window from a host machine, locate and "kill" the process that monitors the operating system.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the computer and actual operating system fault that occurred (Computer will be RED in color).

Test Case 21 Hard Disk Capacity Fault

This test verifies that the System Management Framework tool detects and locates the computer with a storage capacity fault. To simulate full capacity on a storage device, the tester will store large postscript or HDF files to the storage device. This test is limited to workstations and PCs with less than 1 GB of storage. (Test will be verified on all workstations/PCs at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Tester will store as many large postscript and HDF files to the system storage device as possible without damaging any existing files on the system.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the computer (indicated by color of RED) that contains the storage device fault. A message indicating storage capacity should also be displayed.

Test Case 22 Memory Threshold Fault

This test verifies that the System Management Framework tool detects and locates the computer with the memory threshold fault. Since most computers utilize a portion of their hard disk as virtual memory, the Management Information Base object created to monitor a computer's memory will contain threshold limits to determine when the memory capacity of a computer is in excessive use. (Test will be verified on all workstations/PCs at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Tester will execute an application that exceeds the memory threshold limit determined within the Management Information Base object created to monitor the computer memory usage. (If threshold limit is too high to exceed, lower the threshold as needed. After test, threshold limit will be returned to original state.)

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the computer (indicated by color of RED) that contains the memory threshold fault.

Test Case 23 CPU Threshold Fault

This test verifies that the System Management Framework tool detects and locates the computer where the CPU threshold limit has been exceeded. The Management Information Base object created will contain threshold limits to determine when the CPU load of a computer is in excessive use. (Test will be verified on all workstations/PCs at each site.)

Input: Root map of System Management Framework window will be active in the tester's display. Tester will execute an application that utilizes all/most of the CPU processing time.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the computer (indicated by color of RED) where the CPU threshold limit has been exceeded.

Test Case 24 — Multi-Process Termination Fault

This test verifies that the System Management Framework tool detects and locates the computer and processes that have been abruptly terminated. In this test many processes will be activated (GUIs, ftps, etc..), then some of the processes will be abruptly 'killed' simulating software failures.

Input: Root map of System Management Framework window will be active in the tester's display. Activate many processes on a host machine. (FTPs, Telnet, GUIs, etc.) 'Kill' a quarter of the processes from the machine.

Output: Internet symbol on Root map of System Management Framework has turned YELLOW.

Expected Results: Traversing through the Internet submaps, following the YELLOW/marginal status symbols, the tester should be directed to the computer and processes that were terminated.

4.2.6.2.2 Sequence 2 - Policies and Procedures

The following series of tests verify that established management policies and procedures are up-to-date and maintained from site to site. The policies and procedures will be maintained and updated on an as needed basis. The management policies and procedures located at the EDF should be the most current, and all other sites copy must be consistent with that at the EDF.

Test Case 1 — Fault Management

This test verifies that the fault management policies and procedures located at GSFC, MSFC, and LaRC are current and up-to-date with the current at the EDF.

Input: Current fault management policies and procedures from the EDF. Fault management policies and procedures from GSFC, MSFC, and LaRC.

Output: Fault management policies and procedures from GSFC, MSFC, and LaRC.

Expected Results: Fault sections within all documents are current and identical to that copy held at the EDF.

Test Case 2 — Security Management

This test verifies that the security management policies and procedures located at GSFC, MSFC, and LaRC are current and up-to-date with the current from the EDF.

Input: Security management policies and procedures from the EDF.

Output: Security management policies and procedures from GSFC, MSFC, and LaRC.

Expected Results: Security sections within all documents are current and identical to that copy held at the EDF.

Test Case 3 Local Site Management/Security Policy and Procedures

This test verifies that local site security policy & procedures including password management, operational security, data classification, compromise mitigation and access/privileges, systems hardware and software maintenance, and spares inventory guidelines are current and updated at each site supported by this release.

Input: Security management policies and procedures.

Output: Security management policies and procedures from GSFC, MSFC, and LaRC.

Expected Results: Security sections within all documents are current and identical to that copy held at the EDF.

4.2.6.2.3 Sequence 3 - System Network Statistics

The following series of tests verify that the System Management Framework tool is able to perform generation, collection, store and display of system network statistics of the supported sites at the EDF.

Test Case 1 Generation of Network Statistics

This test verifies that the System Management Framework tool is able to generate network statistics.

Input: Request to display network statistics for GSFC, MSFC, LaRC & EDF systems.

Output: Network statistics are displayed for GSFC, MSFC, LaRC & EDF systems.

Expected Results: Network statistics are displayed for DAACs systems are identical to EDF.

Test Case 2 Collection of Network Statistics

This test verifies that the System Management Framework tool is able to collect network statistics.

Input: Active system. System Management Framework window displayed on tester's machine.

Output: Network statistics are displayed for GSFC, MSFC, LaRC & EDF systems.

Expected Results: Network statistics are displayed for DAACs systems are identical to EDF.

Test Case 3 Storing Network Statistics

This test verifies that the System Management Framework tool collects and stores network statistics.

Input: Network statistics for GSFC, MSFC, LaRC & EDF systems.

Output: Network statistics are stored for GSFC, MSFC, LaRC & EDF systems.

Expected Results: Network statistics are displayed for DAACs systems are identical to EDF.

Test Case 4 Display of Network Statistics

This test verifies that the System Management Framework tool accurately displays the network statistics gathered.

Input: Request to display network statistics for GSFC, MSFC, LaRC & EDF systems.

Output: Network statistics are displayed for GSFC, MSFC, LaRC & EDF systems.

Expected Results: Network statistics are displayed for DAACs systems are identical to EDF.

4.2.6.2.4 Sequence 4 - Office Automation Tools

This series of tests demonstrate that a basic set of office automation tools have been installed to support the operational policies and procedures and day-to-day activities and report generation at the DAACs.

Test Case 1 Create Microsoft Word File

Demonstrate that software is installed so files containing the inventory information of the hardware and software can be maintained. That correspondence necessary for communication of status and report generation can be performed.

Input: Microsoft Word software

Output: Microsoft Word software file location

Expected Results: Software is installed on system.

Test Case 2 Create Excel Spread Sheet

Demonstrate that software is installed so files containing the inventory information of the hardware and software can be maintained.

Input:	Excel software
Output:	Excel software file location
Expected Results:	Software installed on system.

4.2.6.2.5 Sequence 5 - M&O Interface for System Administration Management

This series of tests will demonstrate interfaces exist at the DAACs to support various ECS System Administrator functions and responsibilities. This will be performed for all DAAC sites supported for Interim Release 1.

Test Case 1 Active DAAC ECS Administrator Account

An active Administrator account exists to provide a maintenance and operational interface to the DAACs to allow resource usage and management.

Input:	Account with special accesses assigned to System Administrator.
Output:	Account exists with necessary privileges.
Expected Results:	Interface account for system exists.

Test Case 2 ECS Software Backup Maintained

A minimum of one backup save set is maintained in a separate physical location of the ECS software.

Input:	Management policies and procedures guidelines from the EDF.
Output:	Management policies and procedures manual from GSFC, MSFC, and LaRC.
Expected Results:	Section within document are current and identical to that copy held at the EDF.

Test Case 3 Monitoring and Replenishment of Spares Inventory

Verify the monitoring of usage and replenishment of the computer paper, tapes, disks inventory. Guidelines are defined in the Policies and Procedures Manual.

Input:	Management policies and procedures guidelines from the EDF.
Output:	Management policies and procedures manual from GSFC, MSFC, and LaRC.
Expected Results:	Section within document are current and identical to that copy held at the EDF.

4.3 Interim Release 1 Interface Testing

This section of the SITP will test the ECS interface requirements for Interim Release 1. This section is divided into two subsections: External Interfaces and Internal Interfaces. The Internal Interface section will test the ability of the various subsystems of the ECS program to interface with each other.

4.3.1 External Interfaces

As described earlier in this section, TRMM is a platform scheduled for launch in August 1997 which relies on ECS to support its mission. Driven by the launch data but prior to it, and per the EOS Ground System Integration Plan, some ECS capabilities will be available in Interim Release 1. They include:

- Visible Infrared Scanner (VIRS) data products transfer between TRMM Science Data and Information System (TSDIS) and Goddard Space Flight Center (GSFC)
- Precipitation Radar (PR) and TRMM Microwave Image (TMI) data products transfer between TSDIS and Marshall Space Flight Center (MSFC)
- Clouds and Earth’s Radiant Energy System (CERES) and non-TRMM data transfer from TSDIS to Langley Research Center (LaRC)
- CERES Level 0 (L0) data transfer from TRMM SDPF to LaRC
- Lightning Imaging Sensor (LIS) L0 data transfer from TRMM SDPF to MSFC

To support TRMM data transfer and early interface testing, basic ingest services will be available at GSFC, MSFC, and LaRC interfacing with TSDIS and SDPF. Early interface testing between Science Computing Facilities (SCFs) and ECS (GSFC, MSFC, and LaRC) interfaces will be available in order to transfer algorithms and algorithm support data.

Simulators for external interfaces generate and transmit data streams in the identical format that represent the specifics of the actual system’s data stream. However, when the external system becomes available, the actual interface will be utilized. Each element-to-external interface will be tested according to the Interface Requirement Document (IRD) that describes the interface. All modes of data exchange for each interface will be tested to the extent for which they are defined in the IRD.

4.3.2 Internal Interfaces

The major internal interfaces for Interim Release 1 are within SDPS. PGS toolkit deliveries must be made twelve (12) months prior to the Beta reviews for each EOS AM-1 algorithm and twelve (12) months prior to Version 1 delivery for TRMM algorithm, full PGS toolkit support of TRMM data will be available the end of 1994, and Algorithm I&T support for TRMM and EOS AM-1 will be available the end of 1995. The verification of inter-segment and element interfaces is achieved during the execution of the build and thread tests.

Appendix A is in a separate file.

Abbreviations and Acronyms

ADC	Affiliated Data Center
ANSI	American National Standards Institute
APIs	Application Program Interfaces
bpi	bits per inch
CDF	Common Data Format
CDR	Critical Design Review
CD	Compact disk
CDRL	Contract Data Requirements List
CDS	Cell Director Service
CERES	Clouds and Earth's Radiant Energy System
CI	Configured Item
CM	Configuration Management
COTS	Commercial Off the Shelf
CPU	Central Processing Unit
CRR	Capabilities and Requirements Review
CSC	Computer Software Component
CSCI	Computer Software Configuration Item
CSMS	Communications and System Management Segment
CSR	Consent to Ship Review
CSU	Computer Software Unit
DAAC	Data Analysis and Archive Center
DADS	Data Archive Distribution System
DAN	Data Availability Notice
DCE	Distributed Computing Environment
DCN	Document Change Notice
DFS	Distributed File Service
DID	Data Item Description
ECS	EOSDIS Core System

EDC	EROS Data Center
EDF	ECS Development Facility
EOC	Earth Observation Center (Japan); EOS Operations Center (ECS)
EOS	Earth Observing System
EOSDIS	Earth Observing System Data Information System
EPs	Evaluation Packages
ESDIS	Earth Science Data and Information System
ESN	EOSDIS Science Network
ETR	Element Test Review
F&PRS	Functional and Performance Requirements Specification
FOS	Flight Operations Segment
FTP	File Transfer Protocol
GFE	Government furnished equipment
GSFC	Goddard Space Flight Center
GUI	Graphical User Interface
HDF	Hierarchical Data Format
I/O	input/output
I&T	Integration & Test
I&TT	Integration & Test Team
IATO	Independent Acceptance Test Organization
ICD	Interface Control Document
IMS	Information Management System
IR-1	Interim Release 1
IRD	Interface Requirements Document
IV&V	Integration, Verification and Validation
JPL	Jet Propulsion Laboratory
LAN	Local Area Network
LaRC	Langley Research Center
LIS	Lightning Image Sensor
MIB	Management Information Base
M&O	Maintenance and Operations

MSFC	Marshall Space Flight Center
NASA	National Aeronautics and Space Administration
NCR	Non-Conformance Report
NESDIS	National Environmental Satellite, Data and Information Service
NMC	National Meteorological Center (NOAA)
NOAA	National Oceanic and Atmospheric Administration
NSI	NASA Science Internet
ORNL	Oak Ridge National Laboratory
PAIP	Performance Assurance Implementation Plan
PDR	Preliminary Design Review
PDPS	Planning and Data Processing System
PGE	Product Generation Executive
PGS	Product Generation System
POSIX	Portable Operating System Interface for Computer Environment
PR	Precipitation Radar (TRMM)
RMA	Reliability, Maintainability, Avaliability
RPC	Remote Procedure Call
RRR	Release Readiness Review
RTM	Requirements Tracibility Matrix
S/E	Systems Engineering
SCF	Science Computing Facility
SDPF	Sensor Data Processing Facility (GSFC)
SDPS	Science Data Processing Segment
SDR	System Design Review
SI&T	Systems Integration and Test
SITP	System Integration Test Plan
SMC	System Management Center
SRR	System Requirements Review
TBD	To Be Defined/Determined
TCP/IP	Transmission Control Protocol/Internet Protocol
TMI	TRMM Microwave Image

TRMM	Tropical Rainfall Measuring Mission
TSDIS	TRMM Science Data and Information System
VIRS	Visible Infrared Scanner (TRMM)
WAN	Wide Area Network

Glossary

acceptance testing

Verification that is conducted to determine whether a release satisfies its acceptance criteria and that provides the Government with information for determining whether the release should be accepted. Acceptance testing also applies to toolkits, science algorithm integration, and unit-level verification of COTS products.

activity

A specified amount of scheduled work that has a defined start date, takes a specific amount of time to complete, and comprises definable tasks.

affiliated data center (ADC)

A facility not funded by NASA that processes, archives, and distributes Earth science data useful for global change research, with which a working agreement has been negotiated by the EOS program. The agreement provides for the establishment of the degree of connectivity and interoperability between EOSDIS and the ADC needed to meet the specific data access requirements involved in a manner consistent and compatible with EOSDIS services. Such data-related services to be provided to EOSDIS by the ADC can vary considerably for each specific case.

algorithm

Software delivered to the SDPS by a science investigator (PI, TL, or II) to be used as the primary tool in the generation of science products. The term includes executable code, source code, job control scripts, as well as documentation.

analysis

Technical or mathematical evaluation based on calculation, interpolation, or other analytical methods. Analysis involves the processing of accumulated data obtained from other verification methods.

auxiliary data

Auxiliary data can be any data set which enhances the processing or utilization of satellite remote sensing instrument data. The auxiliary data is not captured by the same data collection process as the instrument data. Auxiliary data sets can include data collected by any platform or process, preferably in georeferenced digital format (CEOS).

availability

A measure of the degree to which an item is in an operable and committable state at the start of a "mission" (a requirement to perform its function) when the "mission" is called for an unknown (random) time. (Mathematically, operational availability is defined as the mean time between failures divided by the sum of the mean time between failures and the mean down time [before restoration of function].

baseline

Identification and control of the configuration of software (i.e. selected software work products and their descriptions) at given points in time.

baseline, *configuration management specific*

A configuration identification document or a set of such documents formally designated by the Government at a specific time during a configuration item's life cycle. Baselines, plus approved changes from those baselines, constitute the current approved configuration identification.

build

An assemblage of threads to produce a gradual buildup of system capabilities.

build,

configuration management specific

An assemblage of threads to produce a gradual buildup of system capabilities. Builds are combined with other builds and threads to produce higher level guilds.

calibration

The collection of data required to perform calibration of the instrument science data, instrument engineering data, and the spacecraft engineering data. It includes pre-flight calibration measurements, in-flight calibrator measurements, calibration equation coefficients derived from calibration software routines, and ground truth data that are to be used in the data calibration processing routine.

Capabilities and Requirements Review (CRR)

Assessment of EOSDIS annual Project level capabilities and requirements. CRR is conducted to determine if ECS is meeting its objectives and provides evolutionary direction for new or modified requirements. Current information is provided on how ECS supports the EOS mission.

catalog system

An implementation of a directory, plus a guide and/or inventories, integrated with user support mechanisms that provide data access and answers to inquiries. Capabilities may include browsing, data searches, and placing and taking orders. The system is a specific implementation of a catalog service.

commercial off-the-shelf (COTS)

“Commercial off-the-shelf” means a product, such as an item, material, software, component, subsystem, or system, sold or traded to the general public in the course of normal business operations at prices based on established catalog or market prices.

component

The next lower functional subdivision below "subsystem" in the ECS functional hierarchy.

comprehensive and incremental scheduling

Two modes of scheduling. Comprehensive scheduling is the automatic scheduling of a full set of events. Incremental scheduling is interactive scheduling of selected events. For example, the initial generation of a schedule might use comprehensive scheduling, while the addition of a single event with the desire to avoid perturbing previously scheduled events might use incremental scheduling.

computer software component (CSC)

A distinct part of a computer software configuration item (CSCI). CSCs may be further decomposed into other CSCs and computer software units (CSUs).

computer software configuration item (CSCI)

A configuration item comprised of computer software components (CSCs) and computer software units (CSUs).

computer software unit (CSU)

An element specified in the design of a computer software component (CSC) that is separately testable. A package of work allocated to an individual member of the software development team (CSUs do not span personnel boundaries).

configuration

The functional and physical characteristics of hardware, firmware, software or a combination thereof as set forth in technical document and achieved in a product.

configuration control

The systematic proposal, justification, evaluation, coordination, approval or disapproval of proposed changes, and the implementation of all approved changes in the configuration of a configuration item after formal establishment of its baseline.

configuration item (CI)

An aggregation of hardware, firmware, software or any of its discrete portions, which satisfies an end use function and is designated for configuration management.

configuration item, *configuration management specific*

An aggregation of hardware, firmware, software or any of its discrete portions, which satisfies an end use function and is designated for configuration control. CIs are those items whose performance parameters and physical characteristics must be separately defined (specified) and controlled to provide management insight needed to achieve the overall end use function and performance.

consent to ship review (CSR)

Review to determine the readiness of a release for transition to sites for integration testing.

correlative data

Scientific data from other sources used in the interpretation or validation of instrument data products, e.g., ground truth data and/or data products of other instruments. These data are not utilized for processing instrument data.

critical design review (CDR)

A detailed review of the element/segment-level design, including such details as program design language (PDL) for key software modules, and element interfaces associated with a release.

critical path

A path on a network which has the greatest negative slack and is the longest path in time through the network.

data archive and distribution system

Included in each DAAC [Distributed Active Archive Center], and responsible for archiving and distribution of EOS data and information.

data availability schedule

Data availability schedule is a schedule indicating the times at which specific data sets will be available from remote DADS, EDOS, the IPs, the ADCs and ODCs for ingestion by the collocated DADS. The schedules are received directly by the PGS.

data center

A facility storing, maintaining, and making available data sets for expected use in ongoing and/or future activities. Data centers provide selection and replication of data and needed documentation and, often, the generation of user tailored data products.

data product

A collection (1 or more) of parameters packaged with associated ancillary and labeling data. Uniformly processed and formatted. Typically uniform temporal and spatial resolution. (Often the collection of data distributed by a data center or subsetted by a data center for distribution.) There are two types of data products:

- Standard–A data product produced at a DAAC by a community consensus algorithm. Typically produced for a wide community. May be produced routinely or on-demand. If produced routinely, typically produced over most or all of the available independent variable space. If produced on-demand, produced only on request from users for particular research needs typically over a limited range of independent variable space.
- Special–A data product produced at a science computing facility by a research status algorithm. May migrate to a community consensus algorithm at a later point. If adequate community interest, may be archived and distributed by a DAAC.

data product levels

- Raw data--Data in their original packets, as received from the observer, unprocessed by EDOS.
- Level 0--Raw instrument data at original resolution, time ordered, with duplicate packets removed.
- Level 1A--Reconstructed unprocessed instrument data at full resolution, time referenced, and annotated with ancillary information, including radiometric and geometric calibration coefficients and georeferencing parameters (i.e. platform ephemeris) computed and appended, but not applied to Level 0 data.
- Level 1B--Radiometrically corrected and geolocated Level 1A data that have been processed to sensor units.
- Level 2--Derived geophysical parameters at the same resolution and location as the Level 1 data.
- Level 3--Geophysical parameters that have been spatially and/or temporally re-sampled (i.e., derived from Level 1 or Level 2 data).
- Level 4--Model output and/or results of lower level data that are not directly derived by the instruments.

Data levels 1 through 4 as defined in the EOS Data Panel Report. Consistent with the CODMAC and ESADS definitions.

data quality request

Data quality request is a request issued by the PGS to a scientist at an SCF to perform QA of a particular product before future processing or distribution. A time window is applied to the request in keeping with the production schedule.

dataset

A logically meaningful grouping or collection of similar or related data.

dataset documentation

Information describing the characteristics of a data set and its component granules, including format, source instrumentation, calibration, processing, algorithms, etc.

demonstration

Observation of the functional operation of the verification item in a controlled environment to yield qualitative results without the use of elaborate instrumentation, procedure, or special test equipment.

design interface

The interaction and relationship of logistics with the system engineering process to ensure that system element influences the definition and design of system elements so as to reduce life cycle costs.

deviation

A written request to depart temporarily (for a specific period of time or number of units) from the authorized baseline requirements (i.e., a temporary or limited waiver).

directory

A collection of uniform descriptions that summarize the contents of a large number of data sets. It provides information suitable for making an initial determination of the existence and contents of each data set. Each directory entry contains brief data set information (e.g., type of data, data set name, time and location bounds).

Distributed Active
Archive Center
(DAAC)

An EOSDIS facility which generates, archives, and distributes EOS Standard Products and related information for the duration of the EOS mission. An EOSDIS DAAC is managed by an institution such as a NASA field center or a university, per agreement with NASA. Each DAAC contains functional elements for processing data (the PGS), for archiving and disseminating data (the DADS), and for user services and information management (elements of the IMS).

DAAC Sites
EDC – EROS Data Center
ASF – Alaska SAR Facility
LaRC – Langley Research Center
GSFC – Goddard Space Flight Center
JPL – Jet Propulsion Laboratory
MSFC – Marshall Space Flight Center
NSIDC – National Snow and Ice Data Center
ORNL – Oak Ridge National Laboratory

DAAC-unique

Functions and capabilities provided by the DAAC beyond those provided by the Core System. The functions will be integrated with ECS via APIs for other similar mechanisms. Examples of DAAC-unique functions include visualization, specialized interfaces, and dataset-unique functionality.

ECS evolutionary
development

The process for delivering and evolving ECS functionality through the used of multiple development tracks and delivery mechanisms. Use of development tracks and delivery mechanisms are tailored to the goals of the particular portion of the system of the system, with an overall goal of providing relatively stable portions of the system in comparison to portions which are rapidly adapting to the system's environment.

ECS project

ECS provides single-point access for a worldwide science community, simultaneous mission management for multiple instruments, and regular production of validated science products using community-supplied algorithms.

ECS-supported

A hardware or software component that conforms to an ESDIS approved set of standards and has been fully tested by the ECS contractor.

element

The next lower functional subdivision below "segment" within the ECS functional hierarchy.

element test review (ETR)

Determines if development level testing (for each release) has successfully been completed.

evaluation package

An evaluation package is a delivery mechanism for incrementally developed components and selected prototypes. The objectives of evaluation packages are to increase user involvement in system evolution and rapid evaluation and to facilitate rapid incorporation of user feedback into the incremental development process.

formal development track

A development process distinguished by complete tree of requirements documentation, formal reviews at major milestones in the development cycle and a single waterfall of phases leading to a formal release. The single waterfall has a long time frame relative to the incremental development track and prototypes.

formal qualification testing

A process that allows the contracting agency to determine whether a configuration item complies with the allocated requirements for that item.

formal release

A formal release (Release) is a system-wide update to the ECS, delivered and tested as a part of the EOSDIS. ECS Releases will represent the ECS portion of EOSDIS Versions. Formal releases are part of the formal development track.

function

The action or actions which an item is designed to perform.

functional baseline

The initial baseline established at the system requirements review (SRR) and refined at the system design review (SDR).

functional configuration audit (FCA)

The formal examination of functional characteristics of a CI, prior to acceptance, to verify that the item has achieved the performance specified in its functional or allocated configuration identification.

granule

The smallest aggregation of data that is independently managed (i.e., described, inventoried, retrievable). Granules may be managed as logical granules and/or physical granules.

granule location

The name of the product where this granule is located.

hardware

That combination of subcontracted, commercial-off-the-shelf (COTS), and government furnished equipment (GFE) (e.g., cables and computing machines that are the platforms for software)

hardware configuration item (HWCI)

A configuration item comprised of hardware components.

incremental design review

Review conducted to evaluate segment designs associated with a release.

incremental development track

A development process distinguished by multiple iterations of requirements, design, and implementation with frequent user evaluations via demonstrations. Documentation and reviews are streamlined. Documentation of non-mission critical is created after development has completed. Each increment is developed with the potential of being integrated into the formal track for a release. The incremental development track has a cycle time between the formal development and prototypes.

incremental scheduling

See "comprehensive and incremental scheduling.

independent verification and validation (IV&V)

Verification and validation performed by a contractor or government agency that is not responsible for developing the product or performing the activity being evaluated. IV&V is an activity that is conducted separately from the software development activities governed by the ECS contract.

in operations (or operational)

An ECS capability is in operations if some of the following are true:

- It is accessible by non-ECS personnel (no matter how restricted the group may be)
- Some or all of it resides outside of the EDF
- It is used to support outside agencies/projects (including testing of interfaces with such)

Operations does not necessarily imply responsibility of the formal M&O organization. For example, the early EPs are operated by a combination of the ECS developers and the site liaisons.

inspection

The visual, manual examination of the verification item and comparison to the applicable requirement or other compliance documentation, such as engineering drawings.

instrument

A hardware system that collects scientific or operational data.

Hardware-integrated collection of one or more sensors contributing data of one type to an investigation.

An integrated collection of hardware containing one or more sensors and associated controls designed to produce data on/in an observational environment.

instrument data

Data specifically associated with the instrument, either because they were generated by the instrument or included in data packets identified with that instrument. These data consist of instrument science and engineering data, and possible ancillary data.

instrument science data

Data produced by the science sensor(s) of an instrument, usually constituting the mission of that instrument.

integration

The orderly progression of combining lower level software and/or hardware items to form higher level items with broader capability.

interface(s)

The functional and physical characteristics required to exist at a common boundary.

interim release

The delivery of system capability resulting from early efforts on the formal track development to the customer for testing of EOS functionality prior to an operational version.

interoperability

Refers to the capability of the user interface software of one data set directory or catalog to interact with the user interface at another data set directory or catalog. Three levels of Catalog Interoperability are recognized:

- Level 1 - Simple network interconnectivity among systems.
- Level 2 - Catalog systems can exchange limited search and user information.
- Level 3 - Catalog systems exchange standard search protocols.

This provides "virtual" similarity between different systems.

inventory

A uniform set of descriptions of granules from one or more data sets with information required to select and obtain a subset of those granules. Granule descriptions typically include temporal and spatial coverage, status indicators, and physical storage information. An inventory may describe physical granules, logical granules, or both, including a mapping between them if they are not identical.

Note that the inventory is not the granules themselves, but rather the descriptive data for each of them, specifically used by both system and user to locate those desired.

maintainability

The measure of the ability of an item to be retained in or restored to a specified condition when maintenance is performed by personnel having specified skill levels, using prescribed procedures and resources, at each prescribed level of maintenance and repair. (The probability that maintenance, both corrective and preventive, can be performed in a specified amount of time using a specified set of prescribed procedures and resources expressed as MTTR). Maintainability is the function of design.

maintenance

The process of planning and executing life cycle maintenance concepts and requirements necessary to ensure sustained operation of system elements.

maintenance downtime

The total elapsed time required to repair and restore a system to a full operational status and/or retain a system in that condition.

mean time between failure (MTBF)

The reliability result of the reciprocal of a failure rate that predicts the average number of hours that an item, assembly or piece part will operate within specific design parameters. (MTBF=1/(l) failure rate; (l) failure rate = # of failures/operating time.

mean time between maintenance (MTBM)

The average time between all maintenance including both corrective and preventive maintenance.

mean time to repair (MTTR)

The mean time required to perform corrective maintenance to restore a system/equipment to operate within design parameters.

metadata

Information about data sets which is provided to the ECS by the data supplier or the generating algorithm and which provides a description of the content, format, and utility of the data set. Metadata may be used to select data for a particular scientific investigation.

network

A flow diagram depicting the time phased sequence and interrelationship of events and activities that must be accomplished to achieve project objectives.

nonconformance

The failure of a unit or product to conform to specified requirements.

off-line

Access to information by mail, telephone, facsimile, or other non-direct interface.

on-line

Access to information by direct interface to an information data base via electronic networking.

operational data

Data created by an operational instrument (i.e., NOAA AMRIR).

organizational breakdown

A functionally oriented pyramid-like structure indicating Structure organizational relationships and used as the framework for the assignment of responsibility.

parameter

The output generated by applying predetermined transformation algorithms to previously existing products and ancillary data, using specified calibration coefficients, to represent a specific geophysical parameter. Included are level 2-4 products.

physical configuration audit

The formal examination of the "as-built" configuration of a configuration item against its technical documentation to establish the CI's initial product configuration identification.

Preliminary Design Review (PDR)

PDR is held for each ECS Segment. The PDR addresses the design of the segment-level capabilities and element interfaces through all ECS releases. The PDR also addresses prototyping results and how the results of both Contractor and Government prototyping efforts, studies, and user experience with EOSDIS Version 0 have been incorporated into the ECS design for each respective Segment.

process

A logical sequence of tasks by which a job is accomplished.

product baseline

The baseline which establishes the "as-built" configuration for system-level integration and testing (I&T) and independent acceptance testing. This baseline is validated by functional and physical configuration audits, and reviewed and approved by the Goddard Space Flight Center (GSFC) as part of release readiness review.

product generation executive (PGE)

Product generation executive (PGE) is a set of one or more compiled binary executables and/or command language scripts; it is the smallest schedulable unit for PGS processing.

product generation executable

Product generation executable is an obsolete term. See product generation executive.

product order

Product order is either a request for the generation of a specific product with an associated time window, a priority processing request, a reprocessing request, or a standing order for a product to be generated on a regular basis with a rough timeline, or changes to standing orders. Product orders are received by the PGS from the IMS.

prototyping

The construction of a solution of a design or implementation problem, the feasibility of which needs to be determined as early as possible in order to arrive at a critical decision.

quality assurance

A subset of the total performance assurance activities generally focused on conformance to standards and plans.

reconfiguration

A change in operational hardware, software, data bases or procedures brought about by a change in a system’s objectives.

release initiation review (RIR)

An internal review conducted at the start of the development phase of a release to revisit the requirements and issues associated with that particular release.

release readiness review (RRR)

Conducted at the ECS system level for a GSFC project review team upon completion of release acceptance testing. The IATO leads the RRR to determine, with the GATT and the COTR, if the release is ready to be delivered, installed, and incorporated into the operational system.

regression testing

Re-test to insure the continued correct functioning usually after the replacement or addition of functionality.

requirement

A statement to which the developed system must comply. Varieties of requirements: levels 2, 3, 4; performance, functional, design, interface.

schedules

Schedules represent the current sequence of tasks to be executed along with approximate execution times as generated by the PGS scheduler. Copies of these schedules, which are updated frequently, are made available to the IMS, the SMC, and the DADS.

science computing facility

A facility supplied by the EOS program to an EOS team leader, team member, or principal investigator (instrument or interdisciplinary) for the following purposes: developing and maintaining the algorithms and software used to generate standard data products; quality control of standard data products; in-flight instrument calibration and data set validation; scientific analysis, modeling, and research' generation of special data products; and use as an interface to the investigator's institutional facility.

segment

One of the three functional subdivisions of the ECS, i.e., FOS, SDPS, and CSMS.

simulated data

simulated data - same as test data

software

A combination of associated computer instructions and computer data definitions required to enable the computer hardware to perform computational, data manipulation, and control functions (to include parameters and procedures associated with software products).

software development file

A repository for a collection of material pertinent to the development or support of software. Contents typically include (either direct or by reference) design considerations and constraints, design documentation and data, schedule and status information, test requirements, test cases, test procedures, and test results.

software development library (SDL)

A generic term which describes a controlled collection of software, documentation, and associated tools and procedures used to simplify the development and subsequent support of software. An SDL provides storage of and controlled access to software in both human readable and machine readable form. Also, it may contain management data pertinent to the software development project.

special data products

Data products which are considered part of a research investigation and are produced for a limited region or time period, or data products which are not accepted as standard products.

specification

A document intended primarily for describing the essential technical requirements for items, material or services, including the procedures for determining whether or not the requirements have been met.

standard products

(a) Data products generated as part of a research investigation, of wide research utility, accepted by the IWG and the EOS Program Office, routinely produced, and in general spatially and/or temporally extensive. Standard Level 1 products will be generated for all EOS instruments; standard Level 2 products will be generated for most EOS instruments.

(b) All data products which have been accepted for production at a PGS, including (1) above as well as prototype products.

statement of work

A description of the end objectives, scope, and constraints of a unique and separately identifiable portion of the work required to satisfy contract requirements.

status

Status is information regarding schedules, hardware and software configuration, exception conditions, or processing performance. This information is exchanged with the DADS, and is provided to the SMC. The SMC may also receive information regarding schedule conflicts that have not been resolved with the IMS.

subsampling

Standard subsampling involves extraction of a multi-dimensional rectangular array of pixels from a single data granule, where regularly-spaced, non-consecutive pixels are extracted from each array dimension. For each dimension, the size of the pixel array is characterized by the starting pixel location, the number of pixels to extract, and the pixel-spacing between extracted pixels.

subsetting

Standard subsetting involves extraction of a multi-dimensional rectangular array of pixels from a single data granule, where consecutive pixels are extracted from each array dimension. For each dimension, the size of the pixel array is characterized by the starting pixel location and the number of pixels to extract.

subsystem

A combination of sets, groups, etc., which performs an operational function~~al~~ within a system and is a major division of a system.

support equipment

All equipment required to support the operation and maintenance of a material system. This includes associated multi-use end items, ground handling and maintenance equipment, tools, calibration equipment, communications resources, test equipment and automatic test equipment with diagnostics software for both on-and-off equipment maintenance. It also includes the acquisition of logistics support for the support and test equipment itself.

system

A composite of equipment, skills, and techniques capable of performing or supporting an operational role (or both). A complete system includes all equipment, related facilities, material, software, services and personnel required for its operation and support to the degree that it can be considered a self-sufficient item in its intended operational environment.

System Design Review (SDR)

The SDR addresses the top-level ECS design. The SDR includes the definition and high-level design of ECS segments and elements, the interfaces between these and the interfaces between these and external systems, facilities, users, operators, etc.

System Requirements Review (SRR)

The SRR encompasses a complete review of the ECS specification and the EOS/EOSDIS Requirements (Level 2) that drive the specification, it promotes a common understanding between the Project and the Contractor of the capabilities that ECS must provide.

temporary file

Temporary file is a file which may exists for the duration of a single PGE, or may exist for some indeterminate time beyond the termination of the PGE which created it.

test

A procedure or action taken to determine under real or simulated conditions the capabilities, limitations, characteristics, effectiveness, reliability or suitability of a material, device, system or method.

testing

An element of inspection and generally denotes the determination by technical means of the properties or elements of supplies, or components thereof, including functional operation, and involves the application of scientific principles and procedures.

test data

Test data is any data set designed or specially selected to aid in the algorithm integration and test process, to help test the operation of algorithms under development by simulating realistic input.

test products

Test products are science products generated by new or updated algorithms during the integration and test period. Test products are delivered to scientists at an SCF.

test readiness review (TRR) — Conducted by the project for each release at the segment and element levels to review the plans for the integration and verification of the subsystems with the elements and the elements with their segments.

thread — A set of components (software, hardware, and data) and operational procedures that implement a function or set of functions.

toolkits — Some user toolkits developed by the ECS Contractor will be packaged and delivered on a schedule independent of ECS releases to facilitate science data processing software development and other development activities occurring in parallel with the ECS.

unit — An assembly of any combination of parts, subassemblies and assemblies mounted together which are normally capable of independent operation in a variety of situations.

user — Any person accessing the EOSDIS.

• Authorized users are users who have viable EOSDIS accounts, and who may therefore make EOSDIS data requests. These users may be affiliated or unaffiliated. Affiliated users are those who are sponsored by one of the parties to the Earth Observations-International Coordination Working Group (EO-ICWG) data policy. Each party is responsible for ensuring that all its affiliated users comply with the EO-ICWG data policy. Use of data by affiliated users is classified in one of three categories, defined in the EO-ICWG data policy:

+ Research Use: A study or an investigation in which the user affirms (1) the aim is to establish facts or principles; (2) the data will not be sold or reproduced or provided to anyone not covered by this or another valid affirmation; (3) the results of the research will be submitted for publication in the scientific literature; and (4) detailed results of the research will be provided to the sponsoring spacecraft operator as agreed between the researcher and the sponsoring spacecraft operator. In the context of EOSDIS , this means that NASA-affiliated users must make available to the research community their detailed results, including data, algorithms, and models at the time their research is accepted for publication, and that the data may be copied and shared among other researchers provided that either they are covered by a research agreement or the researcher who obtained the data from EOSDIS is willing to take responsibility for their compliance with the agreement. Data for affiliated users and for research and applications use will be made available at no more than the marginal cost of production and distribution.

+ Environmental Monitoring and Operational Use: Includes data use by those government agencies affiliated with the parties which conduct environmental monitoring and/or operational observations for the public good, and can include larger agencies to which the parties belong (i.e., the World Meteorological Organization); or national agencies, or their designates, involved in other operational forecasting activities which are conducted for the public good (i.e., weather, sea state, sea ice, agriculture, hydrology, etc.). Environmental Monitoring and Operational Use of Data constitutes any use of data to carry out a mandate of environmental observation and prediction as part of an agency's responsibilities to provide for the general welfare. Such use may include the routine downlink or direct broadcast of enhanced and unenhanced data in near-real time within the operational community. Data for Environmental Monitoring and Operational use shall be provided in real or near-real time without fee, and shall be available through international EOS archives for non-real time users for no more than the marginal cost of reproduction and distribution consistent with the access terms for each instrument category.

+ Other users: Those persons requesting data for scientific, operational, applications, or commercial use, who are not directly represented by an EO-ICWG member, and who agree to the stipulations on data access and use as set by the EO-ICWG and the EOS program.

validation

The process of evaluating a system or component during or at the end of the development process to determine whether it satisfies specified requirements.

variance analysis

Identification of variation from a planned baseline and analysis to determine its scope, cause, impact, and corrective action.

verification

The process of evaluating the products of a given development activity to determine correctness and consistency with respect to the products and standards provided as input to that activity.

version

Versions are the culmination of a series of ECS releases, in conjunction with incorporation of SCF-developed science data processing software and unique site capabilities.

waiver, engineering

A written authorization to accept an item, which during manufacture or after having been submitted for inspection, is found to depart from specified requirements, but nevertheless is considered suitable for use “as is” or after repair by an approved method.